MASTER EN SCIENCES INFORMATIQUES

Vision robotique couleur
suivi d'objets en temps réels

Detaille, Octave; Vignoble, Chistophe

Award date:
2006

Facultés Universitaires Notre-Dame de la Paix - Namur
Rue Grandgagnage 21 - B-5000 Namur
Institut d'Informatique

Vision robotique couleur : suivi d'objets en temps réel

Octave Detaille Christophe Vignoble

Mémoire présenté en vue de l'obtention du grade de
Maître en Informatique

Année académique 2005 - 2006

Résumé

Le laboratoire MIPS, de l'Université de Haute-Alsace, en France, souhaite pouvoir effectuer du suivi d'objets en temps réel dans un environnement non contrôlé. A cette fin, le MIPS a développé un algorithme, appelé "Colour Histogram Similarity", permettant de localiser une cible dans des images couleur. Ce mémoire va se concentrer sur l'étude de cet algorithme et de ses limitations dans des environnements non-contrôlés afin de proposer des techniques permettant d'étendre le champ d'application de cet algorithme.

Mots clefs : suivi en temps réel,image couleur, histogramme couleur, similarité, Colour Histogram Similarity (CHS), estimation de taille, redimensionnement, conditions d'illumination, validation

Abstract

The MIPS laboratory, from the University of Haute-Alsace, in France, hope to be able to do some real time tracking in an unknown environment. For this purpose, the MIPS has developed an algorithm, named "Colour Histogram Similarity", allowing to localize a target in a colour image. This master's thesis focus on the study of this algorithm and its limitations in an unknown environment in order to elaborate techniques permitting to spread the field of application of this algorithm.

Keywords : real time tracking, colour image , colour histogram, similarity, colour histogram similarity (CHS), size estimation,resizing ,lighting condition,validation

Nous tenons à remercier les professeurs J.-L. Buessler et J.-P. Urban pour l'accueil qu'ils nous ont réservé au sein du laboratoire MIPS pendant les cinq mois de notre stage. Nous désirons leur faire part de notre gratitude pour le temps qu'ils nous ont consacré et pour les nombreux et précieux conseils qu'ils nous ont fournis afin de guider notre travail à Mulhouse et la rédaction de ce mémoire.

Nous remercions également notre promoteur J.-P. Leclercq pour sa lecture critique et constructive du mémoire, ainsi que pour ses encouragements et son intérêt vis-à-vis de notre travail.

Enfin, nous remercions nos familles pour le soutien qu'elles nous ont apporté tout au long de notre stage et durant la rédaction du mémoire.

Table des matières

Introduction

Dans le cadre d'un projet destiné à prêter assistance aux personnes handicapées, le laboratoire MIPS[1] de l'UHA[2] de Mulhouse s'est intéressé au problème de l'asservissement visuel robotique. Le projet consistait à doter des fauteuils roulants de bras robotisés munis d'une caméra, afin de donner plus d'autonomie à ces personnes. Celles-ci n'auraient qu'à sélectionner, via une interface spécifique, les objets qu'elles désirent manipuler pour que le bras les saisisse automatiquement.

Si un manque de fond a contraint la société en charge du montage des fauteuils d'abandonner le projet, le laboratoire MIPS a poursuivi l'étude de ce problème d'asservissement dans un contexte plus général.

L'asservissement visuel robotique est un processus dans lequel sont en constante interaction deux sous-processus de vision et de commande. Avant d'envisager le pilotage d'un bras, il est indispensable de disposer d'une technique de localisation (vision) des objets précise et fiable, pouvant travailler en temps réel dans un environnement familier.

Pour qu'elle soit applicable dans un maximum de situations, le laboratoire a voulu se limiter à un ensemble minimal d'hypothèses et de contraintes, ce qui rend son approche originale, mais plutôt difficile.

La localisation d'objets en environnement non contrôlé par une caméra en mouvement soulève un certain nombre de difficultés. En particulier, elle doit permettre d'assurer le suivi dans un environnement où les conditions d'illumination sont variables. Aussi, le mouvement du bras sur lequel est monté la caméra induit des variations de taille et de forme considérables de l'image de la cible. Pour assurer une approche fiable et précise, il est nécessaire que la méthode soit tolérante à ces variations ou qu'elle soit capable d'en estimer dynamiquement les paramètres en vue d'effectuer des corrections.

La généralisation de la caméra couleur a incité le laboratoire à opter pour le développement d'une méthode de localisation exploitant la couleur des objets, l'enjeu sous-jacent à son travail étant d'évaluer le potentiel du signal couleur dans des applications de reconnaissance ou de localisation. Parmi le large éventail de techniques de traitement d'images, le MIPS a sélectionné l'approche par intersection

[1] **M**odélisation **I**ntelligence **P**rocessus **S**ystèmes
[2] Université de Haute-Alsace

d'histogrammes dans laquelle les objets sont reconnus sur base de leur distribution de couleurs. Ainsi, il a développé une méthode de suivi temps réel de cible, appelée ***CHS*** (**C**olour **H**istogram **S**imilarity), qui sera l'objet principal de notre étude.

Notre travail fait suite à celui d'André et Frippiat [1] dans lequel ils ont effectué une première évaluation du *CHS* confirmant l'intérêt de l'approche. Cependant celle-ci possède certaines limites qu'il conviendrait de repousser ou d'éliminer pour étendre ses possibilités d'utilisation, ce à quoi nous nous sommes attachés tout au long de notre stage à Mulhouse et que nous présentons dans ce document.

Dans un premier chapitre, nous posons précisément le cadre d'objectifs, de contraintes et d'hypothèses (travail en environnement non contrôlé, robustesse aux variations d'illumination, variation de taille, d'orientation, de perspective, ...) dans lequel le MIPS veut inscrire ses développements en matière de vision. Ceci nous permettra également de situer l'approche du laboratoire par rapport à d'autres approches de vision robotiques.

Dans un deuxième chapitre, nous commençons par présenter brièvement un modèle de formation des images couleurs, dont une compréhension minimale est nécessaire pour aborder le domaine du traitement d'images. Nous y dressons ensuite un état de l'art nous permettant d'une part d'initier le lecteur aux différentes techniques de localisation et de reconnaissance, et d'autre part de consolider le choix du *CHS* par rapport à d'autres approches, ceci en les confrontant au cadre posé précédemment.

Le troisième chapitre est dédié à la présentation et à l'évaluation du *CHS*. Nous y mettons en évidence, tantôt par des réflexions théoriques, tantôt au moyen d'exemples, les limites de l'approche pour définir, ensuite, les aspects qu'il conviendrait de travailler en vue de son amélioration. Nous en profiterons également pour définir et discuter plus précisément les différentes notions sur lesquelles repose le *CHS*.

Dans les chapitres 4 à 6, nous nous attachons à l'étude des plus importants problèmes soulevés lors de la précédente évaluation du chapitre 3. Nous aborderons respectivement les problèmes de sensibilité du *CHS* aux variations d'illuminations, d'estimation de taille et de généralisation au suivi de formes quelconques. A chaque fois, nous avons d'une part exploré les pistes proposées par nos prédécesseurs ainsi que des solutions trouvées dans la littérature scientifique du domaine et d'autre part, nous avons proposé nos propres solutions. Parallèlement à ces travaux, nous avons construit un ensemble d'outils logiciels d'analyse et d'évaluation permettant d'étudier les solutions envisagées ainsi que les théories abordées, et ce, dans un cadre de temps réel. Les plus intéressants de ceux-ci sont brièvement présentés au fil des chapitres.

Dans un dernier chapitre, nous présentons les expérimentations sur prototype robotique réel que nous avons réalisées afin de démontrer l'efficacité et les possibilités des algorithmes que nous avons développés pendant notre séjour à Mulhouse.

Enfin, nous conclurons tout en proposant les perspectives de travail que nous pensons intéressantes à développer dans le futur.

Chapitre 1

Contexte : vision et asservissement visuel

1.1 Introduction

Dans ce chapitre, nous commençons par présenter brièvement l'équipe du laboratoire MIPS qui nous a accueillis durant notre stage en vue de réaliser ce mémoire.

Nous posons ensuite la problématique autour de laquelle va tourner notre étude et définissons un cadre d'hypothèses et de contraintes dans lequel nous voulons inscrire l'ensemble de nos développements.

Nous dressons enfin un état de l'existant afin de situer notre travail par rapport à celui du reste de la communauté du domaine du traitement d'images et définissons, sur base des travaux de nos prédécesseurs, les orientations qui seront les nôtres tout au long de ce mémoire. Nous détaillons également la démarche que nous avons adoptée durant notre séjour à Mulhouse.

1.2 Environnement

Afin de réaliser ce mémoire, nous avons été accueillis au sein du laboratoire **M**odélisation **I**ntelligence **P**rocessus **S**ystèmes (**MIPS**), qui regroupe la recherche de l'**U**niversité de **H**aute **A**lsace (**UHA**) en automatique et ses domaines frontières avec la mécanique, l'informatique, l'optique.

Le laboratoire MIPS se compose de six groupes localisés dans quatre UFR[1] différentes, représentant une soixantaine de personnes (permanents et doctorants). La thématique du MIPS concerne trois sous domaines des STIC[2], l'automatique, le traitement du signal et l'informatique.

Nos travaux s'inscrivent dans un projet de recherche sur l'utilisation de la couleur pour la reconnaissance et localisation de cibles en asservissement visuel. Les travaux

[1] UFR,Unité de Formation et de Recherche
[2] STIC,Sciences et Techniques de l'Information et de la Communication

menés récemment au sein du laboratoire, et par d'autres équipes internationales, confirment la possibilité d'identifier des cibles en se basant sur une signature couleur même en présence de variations d'éclairage.

Afin de valider ses projets, le laboratoire a développé une plate-forme robotique expérimentale composée d'un robot manipulateur six axes, de multiples caméras vidéos, ainsi que d'un réseau d'ordinateurs exécutant des algorithmes de localisation et de commande.

Notre contribution s'inscrit dans ce projet par l'évaluation du potentiel des techniques d'histogrammes couleur pour la phase d'approche dans une tâche d'asservissement visuel.

1.3 Problématique

L'asservissement visuel robotique implique deux processus en constante interaction : l'un de vision, localisant l'objet cible à atteindre, et l'autre de commande, calculant le mouvement du bras robotisé (embarquant éventuellement la caméra).

Afin d'envisager l'asservissement (pilotage) d'un robot, il est important de disposer d'un algorithme de vision fiable et utilisable en temps réel, problématique à laquelle nous nous intéressons tout au long de ce mémoire.

Un peu à l'instar de la vision humaine, certaines techniques de localisation se basent sur l'utilisation de deux caméras pour rendre compte d'une notion de profondeur et ainsi reconstruire l'environnement tridimensionnel observé. On parle alors de vision stéréoscopique.

L'approche qui nous occupe, quant à elle, se base uniquement sur les images délivrées par une seule caméra, ne rendant compte que d'une seule projection de l'environnement, la notion de profondeur pouvant éventuellement être recrée par l'étude des variations des images lorsqu'un mouvement connu est imposé à la caméra (cette dernière question étant indirectement abordée dans le chapitre 5).

Ainsi, le problème de localisation d'un objet dans son environnement se rapporte à un problème de traitement d'images bidimensionnelles qui consiste à en extraire la position de l'image de l'objet (par la suite appelée cible[3] ou image cible) constituant sa projection. Dans le cadre d'une application de suivi temps réel, l'objet ne sera généralement pas représenté par son image de référence[4] mais plutôt par une fonction (notée mod) de celle-ci, calculant ce que nous appellerons par la suite un "modèle de suivi". Il est important d'insister sur le fait qu'il n'est pas question ici de modéliser un environnement tridimensionnel, mais plutôt de travailler sur sa projection sur un plan le long de l'axe optique de la caméra (par la suite appelée scène[5]).

[3] Cible = portion de la scène où se trouve l'image de l'objet (notée m).
[4] Référence = image initiale de la cible servant à calculant de modèle de suivi (notée ref).
[5] Scène = image de l'environnement dans laquelle il faut localiser la cible (noté sc).

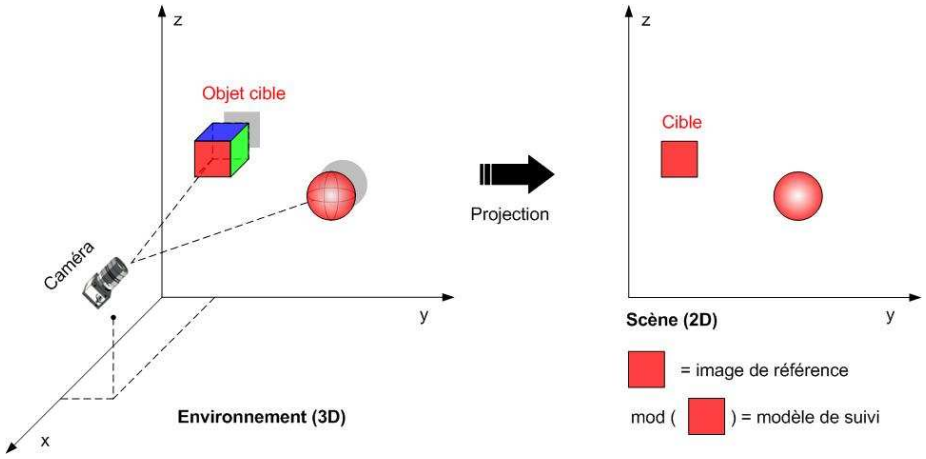


FIG. 1.1 – Projection (à droite) de l'environnement et de l'objet cible (à gauche)

Le suivi de visage [5, 11], de joueurs de football [24, 22], ou la localisation d'objets familiers [7] sont des exemples d'applications de suivi.

Une problème assez similaire au nôtre est celui de l'indexation de bases de données d'images (image retrieval [29, 27]). L'objectif est de permettre la sélection d'une ou plusieurs image(s) la/les plus semblable(s) à une image de référence présentée au système. Bien que les contraintes en jeu dans ce type d'application soient sensiblement différentes des nôtres, notamment au niveau des performances, les techniques y étant développées peuvent être source d'inspiration.

De nombreuses techniques de reconnaissance et de localisation exploitant les images d'intensités (en niveaux de gris) ont déjà été développées, celles-ci donnant déjà d'excellents résultats. Le signal en niveaux de gris se révèle très informatif et permet la détermination de caractéristiques très discriminantes rendant possible l'élaboration de techniques performantes. On peut se demander ce qu'il en est du signal couleur : celui-ci apporte-t-il une réelle information supplémentaire ?

Depuis trois ou quatre ans, nous assistons à un fort engouement pour l'exploitation du signal couleur. La généralisation de la caméra couleur et la puissance sans cesse croissante des ordinateurs étendent désormais le champ du traitement d'images à celui du traitement d'images couleurs. Un des enjeux sous-jacent à cette étude est d'évaluer l'apport de la couleur dans la vision robotique.

Enfin, étant dans un contexte d'asservissement visuel, il est important que les algorithmes soient très rapides. De nombreuses approches fournissent d'excellents résultats en terme de localisation ou de reconnaissance mais avec pour prix un coût calculatoire prohibitif en ce qui nous concerne. Tout au long de cette étude nous mettrons donc l'accent sur la contrainte de temps réel.

Le problème de localisation étant posé, nous pouvons maintenant définir le cadre d'hypothèses et de contraintes dans lequel nous voulons inscrire nos développements.

1.4 Cadre d'hypothése et de contraintes

1.4.1 Hypothèse sur la cible par rapport à l'environnement

Le but poursuivi est de développer un algorithme permettant le suivi de tout type de cibles évoluant dans des environnements très diversifiés et a priori inconnus. Les hypothèses faites sur la cible d'une part, et sur la cible par rapport à son environnement d'autre part, doivent donc être les moins contraignantes possibles. En particulier, le suivi doit pouvoir être assuré sans avoir recours à l'utilisation de marqueurs[6] de couleurs ou de formes spécifiques.

Pour les besoins de l'exposé, appelons *mod* la fonction définie sur l'ensemble des images et à valeur dans l'ensemble des modèles et qui calcule le modèle d'une image.

Le choix de la fonction de modélisation de la cible et de la scène (et comme nous le montrons par la suite, son paramétrage), a un impact direct sur les contraintes imposées sur la cible et son environnement. Cette fonction devrait soit être - idéalement - injective (ie : un même modèle représente au plus une cible), soit telle que le nombre de collisions soit le plus faible possible.

1.4.2 Mouvement relatif de la cible et de la caméra

La cible et son environnement sont perçus par l'intermédiaire d'une projection (le long de l'axe optique de la caméra) sur un plan : le plan de formation de l'image.

De nombreuses approches de localisation et de reconnaissance travaillent avec un modèle d'objet construit sur base d'une seule image, c'est-à-dire construit suivant une projection (ou angle de vue) particulière.

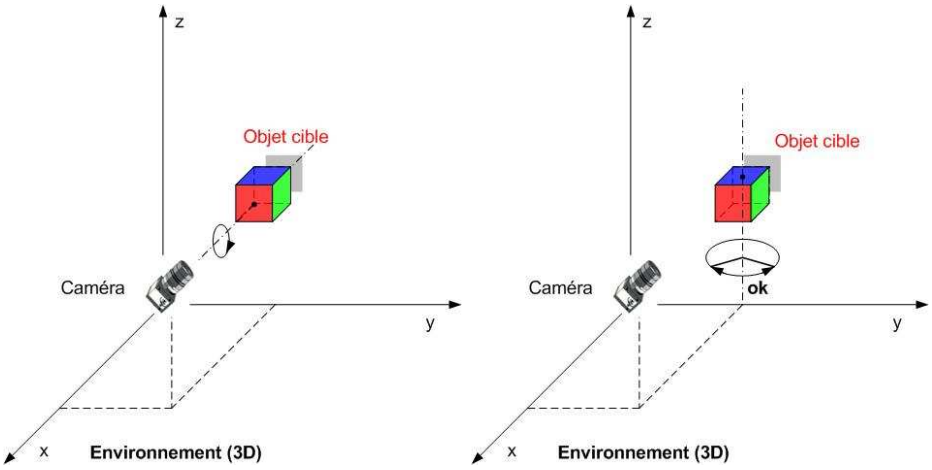
Ceci impose une contrainte sur les mouvements qui modifient l'orientation de l'objet par rapport à la caméra. En effet, le suivi ne peut être assuré que pour les mouvements qui sont tels qu'on puisse, dans la séquence d'images (projections de la scène), retrouver l'image de la cible sur base de laquelle a été calculée le modèle de suivi. La figure 1.2 illustre cette contrainte.

Dans un premier temps, nous adoptons également cette approche, mais il conviendrait par la suite de la généraliser en utilisant par exemple un modèle décrivant la cible sous plusieurs angles de vue afin de s'affranchir de cette contrainte.

1.4.3 Conditions d'illumination

La localisation doit fonctionner même si l'environnement n'est que peu contrôlé. En particulier, une certaine tolérance aux changements de luminosité, tant spectraux[7] que spatiaux[8], sera exigée. Ceux-ci ont pour effet de modifier le spectre réfléchi par l'objet (section 2.2.) et donc l'image qui en est perçue. Une même surface, vue sous

[6] Un marqueur est un élément ajouté à la cible en vue de permettre ou de faciliter son suivi.
[7] Modification du spectre émis par la cible (ex : changement de couleur ou d'intensité)
[8] Changement de position des sources de lumière.



(a) Rotations autour de l'axe optique de la caméra autorisées

(b) Modifications d'orientation limitées par la description du modèle

FIG. 1.2 – Rotation dans le plan et contrainte sur les variations de perspective

deux illuminants différents ne renvoie pas le même spectre et n'est donc pas caractérisée par la même image. L'approche envisagée doit donc pouvoir s'abstraire partiellement ou totalement de ces variations.

1.4.4 Contrainte de temps réel

Le suivi doit pouvoir s'opérer en temps réel. Il ne faut cependant pas perdre de vue que les exigences quant à la cadence de suivi, et donc aussi quant à la dynamique du mouvement, sont inévitablement fonction de la qualité et de la puissance du matériel utilisé. Néanmoins, un suivi à cadence vidéo classique - à savoir 25 images/seconde - devrait pouvoir être assuré sur du matériel de gamme moyenne et donc de coût abordable. Tout au long de cette étude, nous mettons l'accent sur cette composante de temps réel et procédons notamment à des tests de performances. Afin de pouvoir comparer les résultats, ceux-ci sont tous effectués sur une même machine de test[9].

1.5 Point de départ et orientations

Ce mémoire se focalise principalement sur l'étude de la méthode de localisation par **CHS** (**C**olour **H**istogram **S**imilarity, chapitre 3). Initiée par Jean-Luc Buessler et Jean-Philippe Urban [7, 8] dans le cadre de l'asservissement visuel d'un bras robotisé, celle-ci repose sur les travaux de Swain et Ballard traitant de l'indexation de bases de données d'images. Dans le cadre de leur mémoire, André et Frippiat [1] ont plus largement développé cette approche, en s'intéressant notamment à certaines de ses limitations.

[9] Dell Inspiron 8600 - Intel Centrino (Banias) @ 1.7GHz, 512 Mo DDR.

Tous ces travaux confirment l'intérêt de l'approche en matière de localisation. Cependant, l'algorithme de base et les améliorations qui en ont été proposées dans [1] souffrent encore de certaines limitations, les empêchant de rencontrer le cadre d'hypothèses et de contraintes que s'est initialement fixé l'équipe.

Dans un premier temps, nous avons voulu consolider le choix de la méthode du *CHS* pour le suivi d'objet en temps réel dans environnement non contrôlé en dressant un état de l'art et en confrontant systématiquement chacune des techniques y étant répertoriées à notre cadre d'hypothèses et de contraintes (section 2.4).

Par ailleurs, cette étude nous a permis d'entrer dans le domaine du traitement d'images, qui jusqu'alors ne nous était pas familier, et a été source d'inspiration de certains développements ultérieurs, menant à la construction de techniques tirant parti des avantages des différentes techniques.

Nous avons ensuite pris connaissance des problèmes inhérents à la méthode du *CHS*, étudié les solutions apportées par nos prédécesseurs et considéré les perspectives qu'ils proposaient dans leurs conclusions. Lorsque cela s'avérait nécessaire, nous avons également approfondi leurs développements.

Les solutions proposées souffrent encore de certaines limitations. Afin de tenter d'y apporter une réponse, nous avons d'une part exploré la littérature pour adapter et tester certaines approches, et d'autre part, mené nos propres réflexions afin d'apporter des solutions originales, ceci en tenant compte des conclusions de nos prédécesseurs ainsi que de celles auxquelles nous ont mené nos approfondissements.

Parallèlement à ces développements théoriques, nous avons construit un ensemble d'outils Matlab (scripts, librairies et jeux de test) spécifiques à l'étude du *CHS* et à la validation des différentes théories abordées. Nous discutons brièvement de ces implémentations au fil des chapitres de ce mémoire.

Bien qu'une application C++ ait été mise en chantier par Zimmermann [32] et complétée par André et Frippiat [1], nous avons concentré nos efforts sur le développement d'un framework Matlab, à notre sens plus adapté à un contexte de recherche. Matlab dispose d'un ensemble d'outils mathématiques et de traitement d'images qu'il est possible de combiner très facilement pour construire des prototypes expérimentaux. Celui-ci permet également de facilement avoir accès aux résultats intermédiaires et possède bon nombre d'outils permettant de construire des représentations graphiques facilitant l'analyse.

Nous avons également accompli un effort de programmation assez important dans le but d'amener un quasi temps réel sous Matlab. D'une part, nous avons programmé les algorithmes effectuant les traitements coûteux en opérations dans des librairies C optimisées et avons d'autre part défini et implémenté une API permettant d'acquérir des images dans l'espace Matlab à cadence vidéo sur la caméra uEye du laboratoire.

Cet apport nous a permis de démontrer que les algorithmes mis au point étaient

applicables dans un cadre de temps réel. Par ailleurs, l'étude du comportement de ceux-ci sur des séquences vidéos temps réel plutôt que sur des séquences préenregistrées ou des images fixes permettent de mettre en évidence certains problèmes difficilement observables avec les deux premières possibilités.

Pour terminer, tous ces développements et implémentations ont donné lieu, en fin de stage, à la mise en oeuvre d'un prototype robotique réel destiné à valider notre travail (chapitre 7).

Chapitre 2

Techniques de reconnaissance et de localisation de cible

2.1 Introduction

Dans un premier temps, nous commençons par présenter un modèle simplifié du processus de formation de images couleurs sur lesquels reposent les différentes techniques que nous allons aborder.

Nous établissons ensuite un état de l'art (non exhaustif) des techniques envisageables dans le domaine de la localisation et de la reconnaissance. Celui-ci est ensuite confronté au cadre d'hypothèse et de contraintes défini dans le chapitre précédent afin de déterminer les approches étant a priori les plus adéquates en ce qui nous concerne. A l'issue de ce chapitre, nous concluons sur la sélection de l'approche qui sera développée dans le reste de notre étude.

2.2 Formation des images couleurs

Le développement d'algorithmes de traitement d'images ne peut s'envisager sans une bonne compréhension du processus de formations des images. Nous proposons de modéliser la formation d'images comme un processus impliquant trois acteurs : les sources lumineuses, les surfaces, et les observateurs (humains ou électroniques). Les différentes interactions entre ceux-ci sont illustrées à la figure 2.1. La lumière émise par une source lumineuse impacte une surface, avec un certain angle d'incidence, et est réfléchie par celle-ci sur le dispositif optique de l'observateur.

Typiquement, le capteur de celui-ci sépare le spectre en trois gammes de longueurs d'onde correspondant habituellement aux couleurs rouge, verte et bleue pour ensuite les quantifier indépendamment l'une de l'autre. Cette séparation est souvent réalisée à l'aide d'un filtre de Bayer[1]. Chaque point de la scène est alors décrit à l'aide d'un triplet (R,G,B) appartenant à un espace couleur[2] composé des réponses des capteurs. Mathématiquement, les réponses des capteurs peuvent être modélisées [12] par

[1] Filtre de couleur placé devant les capteurs et

[2] Espace engendré par un ensemble de variables colorimétrique ([1], pages 17 à 21) et permettant de caractériser la couleur. RGB en est un exemple, mais ce n'est pas le seul (cfr 3.5.1)

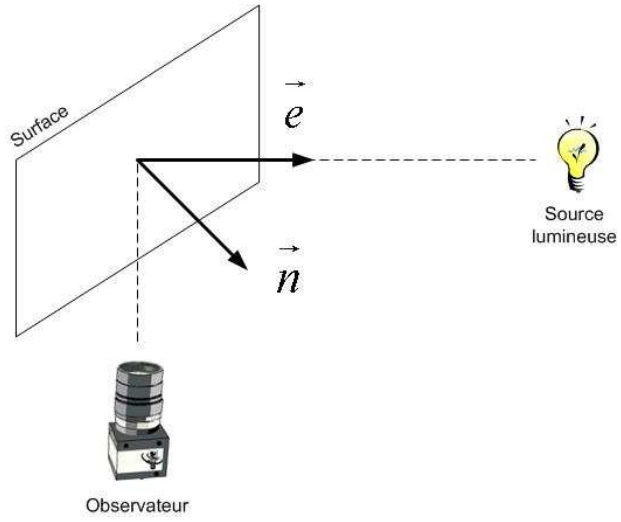


FIG. 2.1 – La lumière émise par la source lumineuse se réfléchit sur la surface pour irradier le capteur de l'observateur

$$R = \int_{\omega} C(\lambda)\ Q_R(\lambda)\ d\lambda \tag{2.1}$$

$$G = \int_{\omega} C(\lambda)\ Q_G(\lambda)\ d\lambda \tag{2.2}$$

$$B = \int_{\omega} C(\lambda)\ Q_B(\lambda)\ d\lambda \tag{2.3}$$

Dans ces équations, Q_R, Q_G et Q_B sont des fonctions de la longueur d'onde caractérisant la réponse des capteurs à un signal couleur incident $C(\lambda)$. Ce signal couleur dépendant lui-même du produit de la lumière incidente à une surface, généralement notée $E(\lambda)$, ainsi que des propriétés de réflectance de cette même surface, caractérisée par la fonction de réflectance $S(\lambda)$:

$$C(\lambda) = E(\lambda)S(\lambda) \tag{2.4}$$

Le signal couleur n'est donc pas une propriété intrinsèque à un objet, mais dépend des conditions dans lesquelles il est perçu. Ainsi, les modifications spectrales de la lumière émise par la source ont un impact direct sur le signal couleur de l'objet, qui doit donc être prudemment utilisé dans l'optique de modélisation d'une cible.

Ces équations constituent un modèle simplifié du processus de formation d'images. Bien que suffisant dans beaucoup de situations, nous proposons d'utiliser une version plus générale, proposée par Finlayson dans [12], qui nous permettra d'aborder les problèmes de dépendance à la configuration géométrique des sources lumineuses. En général, la lumière réfléchie (réflexion spéculaire) par une surface dépend de l'angle entre cette surface et la rayon lumineux qui lui est incident. L'intensité de la lumière réfléchie dépend du cosinus de l'angle entre la normale $\vec{n}$ à la surface (vecteur orthogonal à la surface) et le vecteur d'incidence du rayon lumineux $\vec{e}$.

$$R = (\vec{e} \cdot \vec{n}) \int_{\omega} C(\lambda)\ Q_R(\lambda)\ d\lambda \tag{2.5}$$

$$G = (\vec{e} \cdot \vec{n}) \int_{\omega} C(\lambda)\ Q_G(\lambda)\ d\lambda \tag{2.6}$$

$$B = (\vec{e} \cdot \vec{n}) \int_{\omega} C(\lambda)\ Q_B(\lambda)\ d\lambda \tag{2.7}$$

Ce modèle introduit donc un second type de dépendance du signal couleur, qui est une dépendance spatiale, et qui est liée à la position et à l'orientation relative de la source et de la surface réfléchissante.

Cette dépendance du signal couleur à des conditions externes à l'objet soulève un épineux problème. Un modèle de cible qui serait basé sur l'information couleur ne la caractériserait que dans des conditions particulières ; or le suivi doit pouvoir s'opérer dans un environnement non contrôlé et sur lequel on a donc aucune prise sur les caractéristiques, positions et variations des sources de lumière.

Parallèlement à cette notion physique existe une notion plus "humaine" de la couleur qui est une notion perceptuelle et qu'il importe de ne pas confondre avec la précédente. Ainsi, la couleur est communément définie comme la sensation que produisent sur l'oeil les radiations de la lumière telles qu'elles sont absorbées ou réfléchies par les corps [21]. La perception humaine de la couleur a fait l'objet de très nombreux travaux, en particulier pour assurer une reproduction 'fidèle' (à l'oeil humain) lors de chaînes d'acquisitions et de reproductions (imprimée, TV, ...). Notre souci est ici simplement de caractériser des objets par leurs couleurs, et, en aucun cas, de traiter les images pour les rendre agréables à l'oeil humain.

Afin de caractériser la notion couleur, différentes variables colorimétriques ont étés introduites. Un résumé de celles-ci peut être trouvé dans les pages 17 à 21 de [1].

2.3 État de l'art

Dans cette section nous présentons succinctement les principales techniques de localisation et de reconnaissance que nous avons identifiées. Certaines de ces dernières ont été développées pour résoudre des problèmes différents de la localisation, par exemple l'indexation de grandes bases de données d'images, mais elles sont toutes envisageables dans un cadre de la localisation et de reconnaissance d'objets.

Ceci nous permettra de prendre rapidement connaissance de chacune pour ensuite sélectionner comme point de départ de notre étude celle qui semblera être le plus en adéquation avec l'ensemble de contraintes que s'est initialement fixé le MIPS.

Il existe principalement deux grands types d'approches en matière de localisation. Le premier englobe les techniques exploitant la distribution spatiale des pixels de l'image, tandis que le deuxième reprend celles qui considèrent leur distribution de couleurs. Nous les nommerons respectivement "approches de type géométrique" et

"approches de type distribution de couleurs". Il faut cependant souligner que les premières peuvent également exploiter l'information couleur dans le but de déterminer la structure de l'image.

2.3.1 Approches de type géométrique

Corrélation croisée

Une approche pour localiser des motifs (ou référence dans la terminologie introduite précédemment) dans des images (a)chromatiques, présentée dans [23], consiste à utiliser le coefficient de corrélation croisée. Celui-ci est calculé par produit de convolution de l'image sc et du motif ref (voir figure 2.2). A l'issue de ce processus, une surface de corrélation $s(i,j)$ est obtenue, ses pics représentant les meilleures correspondances. Pour des images en niveau de gris, le coefficient de corrélation croisée se définit de la façon suivante :

$$s(i,j) = \sum_{k=1}^{m}\sum_{l=1}^{n} \left(ref_{k,l} - \overline{ref}\right)\left(sc_{i+k,j+l} - \overline{sc}\right) \tag{2.8}$$

où $\overline{sc}$ et $\overline{ref}$ sont respectivement les moyennes des valeurs des pixels de l'image scène sc, restreinte aux pixels sous le motif, et du motif (de taille $m.n$).

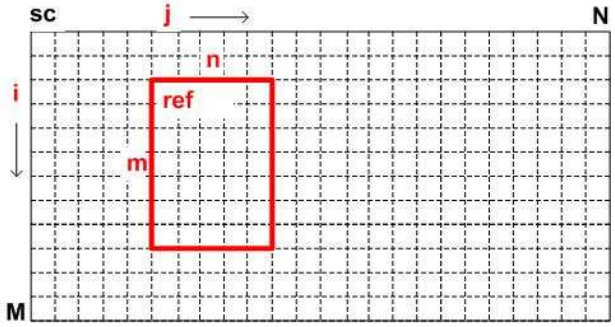


FIG. 2.2 – Pour toutes les positions (i,j) d'une fenêtre de l'image sc, le coefficient de corrélation entre celle-ci et le motif est calculé (produit de convolution).

Hachage géométrique

Le hachage géométrique [31] est une technique qui a été initialement développée pour la vision par ordinateur, et qui tente de faire correspondre les caractéristiques géométriques présentes dans une scène avec celles d'objets préalablement analysés.

Une première phase consiste à construire une table de hachage associant des caractéristiques à des modèles. Celles-ci sont premièrement identifiées dans les différents modèles, et ensuite prises deux à deux pour construire des repères orthogonaux utilisés pour y ré-exprimer relativement les coordonnées des autres caractéristiques. Un ensemble de représentations du même modèle dans toutes les bases (de caractéristiques) possibles est alors décrit par cette table de hachage bidimensionnelle, indexée

par les positions relatives calculées (figure 2.3).

La table de hachage contient pour chaque entrée, un ensemble de couples *(modèle, base)*, chacun de ces couples signifiant que le modèle *modèle* possède une caractéristique à la position donnée par cet index, position exprimée dans une base *base*.

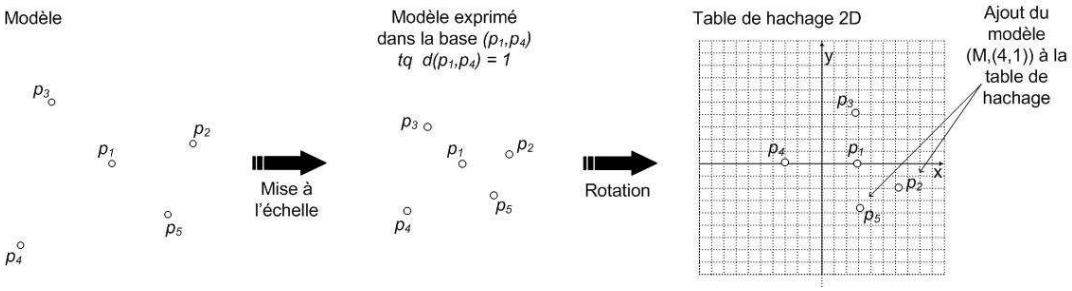


FIG. 2.3 – Construction de la table de hachage indexant les modèles exprimés dans les différentes bases (source [31])

La phase de localisation commence par extraire toutes les caractéristiques présentes dans la scène. Un base est ensuite sélectionnée arbitrairement pour y réexprimer les coordonnées absolues de toutes les autres caractéristiques. Ces coordonnées servent pour accéder à la table de hachage. A chacun des couples (*modèle*, *base*) indexés par ces coordonnées, un vote est attribué. Une fois toutes les caractéristiques prises en compte, les modèles ayant un nombre de votes suffisant sont considérés comme correspondance potentielle. La meilleure représentation (du modèle, dans une base) est sélectionnée par une technique d'optimisation pour ensuite être confrontée à l'image originale pour validation. Si celle-ci échoue, le même procédé est ré-appliqué avec une base différente (figure 2.4).

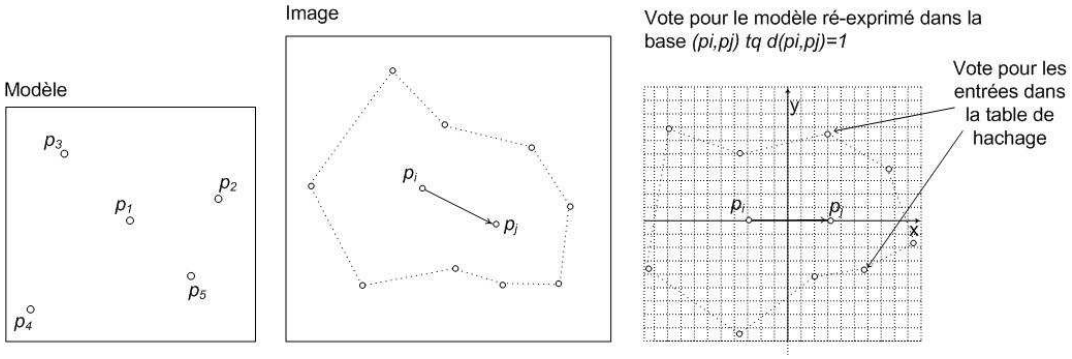


FIG. 2.4 – Processus de localisation par vote sur les modèles, dans la table de hachage (source [31])

Cette méthode a été utilisée pour résoudre des problèmes de "pattern-matching" en vision informatique, pour des logiciels de types CAD/CAM (Computer-Aided Design/Computer-Aided Manufacturing) , et l'ingénierie médicale. Cette technique a été étendue aux modèles contenant des degrés de liberté interne, par exemple un humain qui est alors considéré comme un ensemble articulé d'éléments rigides.

Modèles à contour actif (Snakes)

Introduit formellement par Kass et Witkin en 1987, le modèle à contour actif (snake) est une structure dynamique utilisée en traitement d'images et en vision artificielle. Celui-ci permet de résoudre le problème de la détection de contours en utilisant un modèle de courbe déformable qui épouse la forme des objets.

Un modèle de contour actif est formé d'une série de points mobiles et répartis sur une courbe en deux dimensions (équation 2.9). La courbe, ouverte ou fermée, est placée dans la zone d'intérêt de l'image ou autour d'un objet.

$$v(s) = [x(s), y(s)], s \in [0, 1] \tag{2.9}$$

Plusieurs équations décrivent son évolution : les points de la courbe se déplacent de façon telle qu'elle épouse lentement les contours des objets en fonction de divers paramètres comme l'élasticité, la tolérance au bruit, etc.

Cette dynamique est basée sur la notion d'énergie interne et externe, le but étant de minimiser l'énergie totale présente le long de la courbe. Des contraintes permettent de conserver une courbe lisse avec des points équidistants tout en laissant un certain champ libre pour les déformations. L'énergie interne correspond à la morphologie et aux caractéristiques de la courbe (courbature, longueur, etc.). L'énergie externe provient de l'image, les critères sont variables (présence de bords marqués, bruit, etc.).

L'évolution se fait de manière itérative et les algorithmes peuvent faire l'objet de diverses optimisations et techniques numériques.

(a) Contour initial (b) Contour après optimisation

FIG. 2.5 – Les points de la courbe sont déplacés pour qu'elle épouse les formes de la fleur

Pour plus de détails, on pourra s'en référer à [4] ainsi qu'à [22] dans lequel est présentée une méthode de suivi de joueurs de football en temps réel, et ce grâce à l'utilisation d'algorithmes d'optimisation de type greedy, peu coûteux en temps de calcul.
Dans [22], la localisation initiale de l'objet est réalisée manuellement. Le contour est

ensuite agrandi d'un certain facteur, et celui-ci est utilisé comme contour de base pour l'image suivante de la séquence. L'application de l'algorithme ajuste la courbe au contour de la cible. A l'itération suivante, le contour sera de nouveau agrandi pour être ensuite réajusté sur le contour de la cible et ainsi de suite.

Segmentation

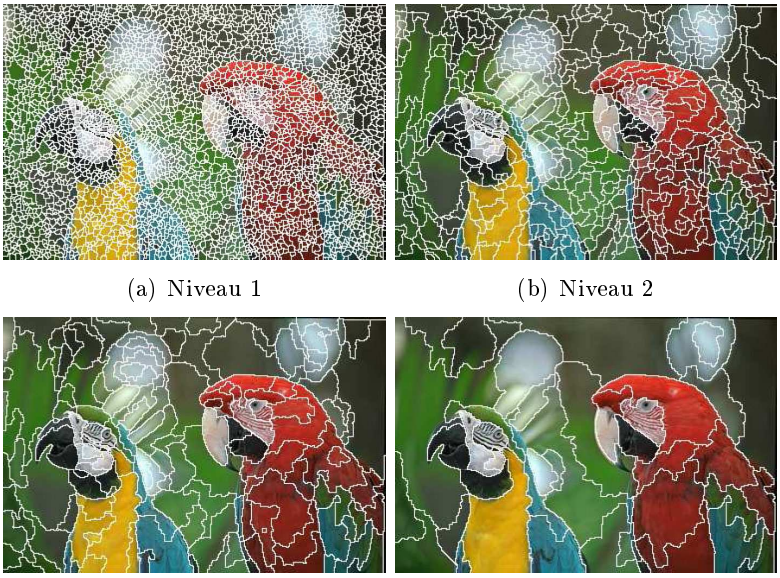
La segmentation consiste à partitionner une image en zones homogènes selon un critère fixé. Ce dernier peut soit être de type géométrique (approche contour) soit de type visuel (approche région). Ce processus de segmentation est très utilisé dans l'analyse d'images, et constitue généralement une étape préliminaire dans les applications de suivi, notamment dans des applications de suivi de visages ou de suivi de joueurs de football. En vue de valider la détection d'une cible au moyen de ses partitions, des mesures statistiques peuvent être calculées (aire, moments d'inertie, ...). Plusieurs méthodes ont été proposées dans la littérature, nous nous limiterons ici à présenter le principe de quelques-unes des plus courantes.

Les algorithmes dits de "division et fusion de région" sont une méthode usuelle pour segmenter une image (ex : quadtree). Elles consistent à considérer une portion de l'image et à vérifier si tous les points de celle-ci satisfont un critère défini. Lorsque ce n'est pas le cas, la région est subdivisée en sous-régions (4 pour le quadtree) et le même principe y est appliqué récursivement. Dans un second temps, les régions connexes satisfaisant ce critère sont ensuite fusionnées.

D'autres algorithmes, dits de "croissance de régions", ont aussi étés utilisés, car les régions trouvées par ceux-ci coïncident généralement avec les frontières observées par l'oeil humain. Depuis un pixel de base (un germe) arbitrairement fixé, la région de ce pixel est agrandie en examinant tous ses voisins et en les ajoutant ou non à la région en fonction d'un critère de similarité donné. Une fois à sa taille maximale, un nouveau point est choisi aléatoirement et le processus est répété. L'algorithme s'arrête lorsque toute la surface de l'image a été couverte. Afin de limiter les impacts (pouvant être néfastes) des choix consécutifs des positions des germes, des variantes faisant croître simultanément plusieurs régions (plusieurs germes) ont été développés.

La LPE (ligne de partage des eaux, figure 2.6) est un outil de segmentation provenant de la morphologie mathématique. Intuitivement, il s'agit de considérer l'image en niveau de gris comme une surface, pour laquelle la valeur de gris correspond à une hauteur. Cette surface est inondée par ses minima locaux, et lorsque deux "lacs" se rejoignent nous trouvons une ligne de partage des eaux, ligne séparant donc les versants (ou zones d'influences des minima). De manière générale, l'algorithme de LPE est appliqué au gradient morphologique de l'image elle-même et non à l'image telle quelle, afin que les bordures des régions coïncident avec les discontinuités. Ainsi, il peut être utilisé aussi bien sur des images couleurs que sur les images en niveau de gris. Plusieurs implémentations de ce principe ont été réalisées, et celles-ci ont été comparées par Roerdink et Meijster dans [25].

La segmentation ne constitue généralement qu'une étape préliminaire dans le processus de localisation. La segmentation seule ne nous suffit pas pour suivre un objet. Cependant, la localisation peut se faire en comparant les régions présentes dans



(a) Niveau 1 (b) Niveau 2

(c) Niveau 3 (d) Niveau 4

FIG. 2.6 – Ligne de partage des eaux (LPE) (source [26])

l'image avec celles de l'objet de référence, et en identifiant les régions correspondant à l'objet cible dans l'image, notamment au moyen de mesures statistiques ou de technique d'appariement de graphes.

2.3.2 Approches de type distribution de couleurs

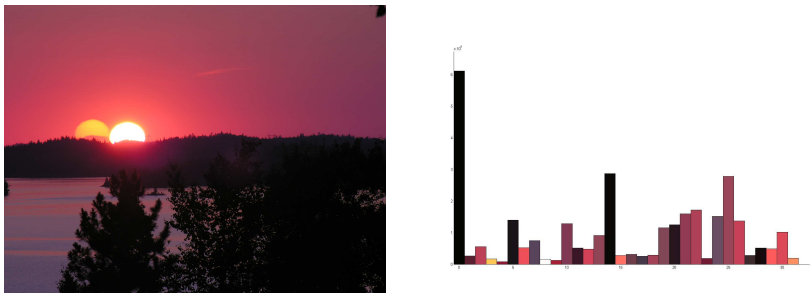
Les techniques suivantes ont déjà été présentées par André et Frippiat dans [1]. Nous limitons donc à en faire une brève description. Avant de commencer, nous définissons brièvement la notion d'histogramme, sur laquelle repose toutes les approches présentées dans cette sous-section.

L'histogramme couleur H est un vecteur représentant la distribution des couleurs d'une image. Chacune de ses P composantes $H(i)$ représente le nombre de pixels dont la couleur est voisine à la couleur c_i lui étant associée. Les $H(i)$ comptabilisent donc les pixels des classes (ou ensembles) C_i de couleurs voisines aux c_i qui les représentent (figure 2.7). Par abus de langage, on fera souvent référence aux composantes de l'histogramme par le terme "classe de l'histogramme". Dans la littérature anglo-saxonne, celles-ci sont généralement appelées "bins".

Rétro-projection d'histogrammes

La méthode par rétro-projection d'histogrammes ("histogram backprojection") a été initialement proposée par Swain et Ballard [29]. Un histogramme ratio R, rapport des histogrammes de la référence H_{ref} et de la scène H_{sc}, est d'abord calculé. A chaque classe de couleur c_i de l'histogramme, on associe le ratio $R(i)$ correspondant au moyen de la formule suivante :

$$\forall i = 1, \ldots, P \quad R(i) = \min\left(\frac{H_{ref}\ (i)}{H_{sc}\ (i)}, 1\right) \tag{2.10}$$



(a) Une image (b) Son histogramme

FIG. 2.7 – Exemple d'histogramme (b) d'une image (a) construit avec 32 classes. La couleur de chaque batôn est la couleur c_i représentative de chaque classe C_i

Chaque pixel est ensuite remplacé par le ratio correspondant à sa classe, qui représente en fait la probabilité que la couleur appartienne à la cible. L'algorithme considère que la position de celle-ci est la zone où la densité calculée par convolution entre un masque et l'image est maximale. Le produit de convolution revient à déplacer sur l'image ratio une silhouette, ie, un masque de la taille et de la forme attendue de la cible, et de calculer la somme des probabilités des éléments de la silhouette pour chaque position de cette dernière dans l'image ratio. La position dont la somme des probabilités donne la plus grande valeur est considérée comme la position réelle de l'objet cible (figure 2.8).

(a) Image originale (b) Backprojection et localisation (cadre vert)

FIG. 2.8 – Image et image rétro-projetée (source [1])

Mean shift

L'algorithme du MeanShift [5] est variante de la rétro-projection d'histogrammes, visant à pallier certains défauts de cette dernière. Sa recherche du maximum plus locale, suivant la descente du gradient, lui permet d'éviter de confondre trop facilement la cible recherchée avec des objets plus éloignés. Cet algorithme est itératif et utilise les moments statistiques d'ordre zéro et d'ordre un afin de recentrer sa fenêtre solution sur le centroïde de la cible.

Soit M_{00} le moment d'ordre 0 :

$$M_{00} = \frac{1}{M.N} \sum_i \sum_j sc(i,j) \tag{2.11}$$

Soient M_{10} et M_{01} les moments statistiques d'ordres premiers selon i et j respectivement :

$$M_{10} = \frac{1}{M.N} \sum_i \sum_j i * sc(i,j) \tag{2.12}$$

$$M_{01} = \frac{1}{M.N} \sum_i \sum_j j * sc(i,j) \tag{2.13}$$

La position du centroïde de la solution lors d'une itération est donnée par :

$$x_c = \frac{M_{10}}{M_{00}} \tag{2.14}$$

$$y_c = \frac{M_{01}}{M_{00}} \tag{2.15}$$

Lorsque le recentrage entre deux itérations de l'algorithme est inférieur à un certain seuil, l'algorithme s'arrête.

Cam Shift

Le Cam shift est une procédure basée sur le Mean Shift, développée par Bradski [5] et visant à le perfectionner. Le Mean Shift amélioré est moins sensible à l'environnement que la rétro-projection, mais ne gère pas la taille et l'orientation de la cible dans l'image. Afin de pallier ce problème, Bradski propose d'utiliser non seulement les moments d'ordre un mais aussi ceux d'ordre deux afin de déterminer la taille et l'orientation de la cible. A chaque itération, la largeur et la hauteur de la cible sont recalculées, de même que l'orientation de la cible. Pour pouvoir envisager un agrandissement de la cible, c'est-à-dire son rapprochement de la caméra, la fenêtre utilisée pour localiser la cible est agrandie d'un certain facteur dépendant de la dynamique de l'objet à localiser. Cette technique est particulièrement adaptée au suivi de visage.

Intersection d'histogrammes

La méthode d'intersection d'histogrammes est une technique de localisation, basée exclusivement sur la couleur, et présentée pour la première fois par Swain et Ballard. Dans [29], ils mettent en évidence la forte capacité des histogrammes couleurs à discriminer les images et proposent de les utiliser en tant que signatures de celles-ci. La comparaison entre images est réalisée en calculant une mesure d'intersection entre histogrammes, variant entre zéro et un, indicative de la similarité. Afin de localiser la cible, on calcule cette mesure pour toutes les positions d'un masque appliqué à la scène. La position de la cible considérée par l'algorithme est celle où la similarité est maximale. Plusieurs variantes de cet algorithme ont été développées, telles l'intersection d'histogrammes tridimensionnels ou monodimensionnels. Nous détaillerons l'adaptation de cette technique au suivi d'objets en temps réel, effectuée par le groupe MIPS, dans le chapitre 3.

2.3.3 Autres techniques

En plus des ces approches plus classiques, il existe de nombreuses méthodes de localisation basées sur des théories plus complexes telles les transformées de Fourier, les ondelettes (technique spatio-fréquentielle) [11], les champs de Markov, les fractales, les matrices de co-occurrence ou d'auto-covariance que nous n'aborderons pas ici, celles-ci étant extrêmement complexes et ne donnant encore que de modestes résultats.

2.4 Evaluation

Comme l'illustre l'état de l'art que nous venons d'établir, les approches en matière de vision sont nombreuses et il nous faudra dès lors en choisir une comme point de départ de nos investigations, a priori la plus en adéquation avec les objectifs et contraintes que s'est fixés le groupe MIPS.

Nous allons nous servir du cadre hypothétique que s'est fixé le MIPS pour argumenter notre choix. Nous allons confronter à ce cadre hypothétique chacune des techniques que nous avons présentées, à l'exception toutefois de la segmentation, car celle-ci ne constitue pas une méthode de suivi à proprement parler, et les résultats obtenus dépendront fortement non seulement de la méthode utilisée pour segmenter mais aussi de la technique utilisée pour comparer les régions de l'image avec celles constituant le modèle.

2.4.1 Hypothèse sur la cible par rapport à l'environnement

La corrélation croisée utilise la fonction identité (bijective) comme fonction de modélisation. Le modèle construit sur l'image de la cible est l'image de la cible elle-même. Il n'y a donc aucun moyen de la confondre avec une zone de la scène en utilisant ce type de modèle.

La fonction de modélisation utilisée dans le hachage géométrique n'est bien sûr pas injective. En effet, deux cibles ne différant que par leurs couleurs comportent les mêmes particularités géométriques (coins, droites, ...) et sont donc représentées par le même modèle. Toutefois, comme montré dans [31], le hachage a une excellente capacité à indexer de larges collections d'objets, ce qui signifie un nombre de collisions réduit pour la fonction de modélisation. Il est donc raisonnable de prendre comme hypothèse que la signature géométrique[3] est suffisamment discriminante pour localiser un objet au sein d'une scène.

Les approches dites par "modèle à contour actif" (ou "Snakes") modélisent la cible à l'aide d'une courbe positionnée sur son contour extérieur. Ces techniques fonctionnent très bien lorsque les couleurs de la cible se détachent d'un fond idéalement uniforme. Elles sont en particulier bien adaptées à des applications du type suivi de joueurs de football [22]. Ces contraintes sont trop fortes et ne peuvent être

[3] Signature géométrique : définition

acceptées dans le cadre que nous nous sommes fixé.

Les approches à base de rétro-projection ("histogram backprojection", "Mean Shift" et "Cam Shift") fonctionnent très bien sous l'hypothèse que les couleurs de la cible ne se retrouvent pas ou presque pas dans l'environnement. Ceci convient parfaitement pour du suivi de visages en temps réel comme l'a montré Bradski dans [5], la teinte de la peau se détachant généralement du fond. Les expérimentations réalisées au MIPS ont montré que ces techniques n'étaient pas satisfaisantes pour le suivi d'objets dans des environnements de la vie quotidienne.

Enfin, les approches basées sur l'intersection d'histogrammes modélisent les cibles à l'aide d'histogrammes couleurs. La fonction de modélisation (la transformée en histogramme) n'est bien sûr pas injective, mais il est cependant raisonnable de considérer que la cible peut être identifiée de manière suffisamment discriminante sur base de sa distribution de couleurs. Swain a en effet montré que celle-ci pouvait très bien être utilisée pour indexer efficacement de larges collections d'images. Soulignons que ceci autorise que les couleurs de la cible se retrouvent dans le fond, tant qu'elles n'y sont pas en même proportion que dans la cible à localiser.

2.4.2 Mouvement relatif de la cible et de la caméra

Si la méthode par corrélation croisée permet de gérer naturellement les translations dans un plan orthogonal à l'axe de caméra, il n'en est ni de même pour les mouvements modifiant la distance cible/caméra, ni pour les rotations. Il est en effet nécessaire de connaître avec précision la taille apparente (variant en fonction de la distance cible/caméra) et l'orientation de la cible avant de la localiser. Ceci est en effet nécessaire afin de pouvoir ajuster le modèle en vue de calculer la surface de corrélation. De plus, une faible erreur dans l'estimation de ces paramètres induit généralement une forte diminution de la valeur du coefficient de corrélation, ce qui rend généralement la détection incertaine.

La technique du hachage géométrique permet, quant à elle, de prendre en charge tous les types de mouvements. Les changements de taille et les rotations sont naturellement gérés par la réexpression des coordonnées des caractéristiques dans les différentes bases associées aux différents modèles. De plus, d'après Wolfson, l'utilisation de particularités suffisamment complexes permettrait de traiter les mouvements de rotation hors du plan, c'est-à-dire des changements de perspective.

Dans les modèles à contour actif, la variation de taille est gérée à l'aide d'une "force ballon" introduite par Cohen dans [10] qui permet au contour de grossir ou de se réduire en fonction de la direction de cette force. La déformation et l'orientation du contour dépendent directement de la fonctionnelle d'énergie. La minimisation de celle-ci le long du contour oriente donc automatiquement le contour autour de la cible.

Tout comme la méthode par corrélation croisée, les approches par rétro-projection d'histogrammes, Mean Shift et d'intersection d'histogrammes nécessitent une estimation a priori de la taille de la cible et de son orientation afin de pouvoir ajuster le

masque et le modèle. Cependant, elles présentent un double avantage sur la première. Les variations de taille de la cible se reflètent par une simple multiplication scalaire de l'histogramme par un facteur d'échelle, ce qui constitue une opération beaucoup plus simple et plus efficace que le re-dimensionnement d'une image bidimensionnelle par exemple. Secondement, les histogrammes ont pour caractéristique de faire complètement disparaître toute information spatiale, de par leur nature statistique (il ne s'agit en effet que d'un dénombrement, ne tenant absolument pas compte des positions des pixels). De ce fait, ce type de modèle a la propriété extrêmement intéressante d'être invariant aux rotations et ne doit donc pas être réajusté avant localisation[4].

Le Cam Shift, amélioration du Mean Shift, permet d'estimer le facteur d'échelle apparent et l'orientation de la cible par calcul des moments statistiques du premier et du deuxième ordre. Cependant, cette technique ne fonctionne que sous la condition très restrictive que les couleurs de la cible ne soient pas ou presque pas présentes dans la scène. Ceci justifie entre autres ses bons résultats dans les applications de suivi de visage où la couleur de la peau ne se retrouve généralement pas dans l'environnement.

2.4.3 Conditions d'illumination

La méthode par corrélation croisée ne présente, telle quelle, aucune tolérance aux changements de luminosité. En effet, étant donné qu'aucune correction n'est apportée (soit du côté du modèle, soit du côté de la scène), le calcul du coefficient de corrélation perd tout son sens et sa valeur n'est dans ce cas plus du tout représentative d'une véritable correspondance.

Le hachage géométrique, quant à lui, peut cependant fournir une tolérance aux changements de luminosité. En effet, si l'on admet le postulat selon lequel l'effet de l'illuminant est le même sur des pixels voisins [15], les particularités géométriques n'en seront pas affectées, celles-ci correspondant généralement à de brusques changements d'intensité.

En acceptant ce même postulat, on est en droit de supposer que les contours resteront des contours, même sous un autre illuminant. En effet, les contours correspondent également à de brusques changements d'intensité, et l'effet local de l'illuminant étant considéré comme constant, ces changements seront conservés. Cependant, un contour est beaucoup moins local qu'une particularité, et l'effet de l'illuminant peut différer d'un point à l'autre du contour. D'autre part, les ombres peuvent conduire à faire apparaître de nouveaux contours ou à en masquer d'autres. Les contours pourront donc être altérés et le suivi dégradé.

Les approches à base de rétro-projection et d'intersection d'histogrammes ne fournissent pas ou presque pas de tolérance aux changements de luminosité. En effet, ce type de changement induit des transferts de pixels entre classes dans ces histogrammes, leur rétro-projection ou leur comparaison étant donc faussée. Cependant, il est envisageable d'effectuer des traitements sur le modèle ou sur la scène afin

[4] L'invariance du modèle à l'orientation est une condition nécessaire pour que la technique y soit aussi invariante, mais elle n'est pas suffisante.

d'accroître la tolérance de ces techniques. Nous reviendrons sur ce problème dans le chapitre 4.

2.4.4 Contrainte de temps réel

Le calcul du coefficient de corrélation est très coûteux en temps s'il est calculé dans le domaine spatial. Cependant, Lewis a montré [23] qu'il était possible de le diminuer en ramenant son calcul dans l'espace des fréquences, à l'aide des transformées de Fourier, rendant ainsi son utilisation envisageable pour de la localisation temps réel.

Le temps de calcul nécessaire à la reconnaissance d'un objet par hachage géométrique est une fonction polynomiale du nombre de particularités dans la scène et de degré supérieur ou égal à quatre, ce qui est trop important pour envisager une application temps réel. Notons aussi que, selon les auteurs, ce calcul est fortement parallélisable. Cependant, cette solution n'est pas non plus envisageable étant donné que nous voulons travailler avec des machines de gamme moyenne, ne possédant en général qu'un seul processeur, de puissance assez limitée, bien que l'avènement du " multi-core " se profile de plus en plus et que les plates-formes multiprocesseurs deviennent de plus en plus abordables.

Les techniques basées sur la segmentation sont inapplicables dans le cadre d'applications temps réel, et ce pour une double raison. La segmentation, opération très coûteuse en temps de calcul, n'est qu'une étape préalable au processus de reconnaissance, pouvant lui aussi être très coûteuse. Nous citerons en particulier le cas des algorithmes d'appariement de graphes, qui sont de forte complexité.

Comme montré dans [22], les techniques basées sur les "snakes" peuvent aisément être adaptées au suivi d'objets en temps réel, et ce grâce à l'applicabilité d'algorithmes d'optimisation de type greedy.

Les techniques de rétro-projection, Mean Shift, et Cam Shift sont parfaitement adaptées à des applications temps réel. En effet, le coût calculatoire nécessaire pour rétro-projeter l'histogramme ratio et pour déterminer le maximum de densité est extrêmement faible.

Les techniques basées sur les intersections d'histogrammes ont également un net avantage en terme de coût calculatoire. En effet, les histogrammes ne constituant qu'un simple dénombrement (après re-quantification de l'espace couleur) se construisent en temps linéaire par rapport à la taille de l'image. De plus, l'algorithme calculant la surface de similarité peut être optimisé pour éviter les calculs redondants et donc réduire fortement sa complexité. Nous discuterons de cette possibilité en 3.4.1

Le tableau suivant met en relation de façon plus synthétique les quatre contraintes imposées sur l'algorithme et les différentes techniques brièvement discutées.

Technique	Cible/Env.	Mvt cible/cam			Rob. lum.	App. temps réel
		Trans.	Rot. (p)	Rot. (hp)		
Corrélation croisée	OK	KO (ech)	KO	Tolérant	Très faible	OK
Hachage géométrique	OK	OK	OK	OK	OK	KO
Contours actifs	KO	OK	OK	Tolérant	Tolérant	OK
Back projection	KO	KO (ech)	Tolérant	Tolérant	KO	OK
Mean Shift	KO	KO (ech)	Tolérant	Tolérant	Très faible	OK
Cam Shift	KO	OK	OK	Tolérant	Très faible	OK
Intersection d'hist. (1D)	OK	KO (ech)	Tolérant	Tolérant	Très faible	OK
Intersection d'hist. (3D)	OK	KO (ech)	Tolérant	Tolérant	KO	OK

TAB. 2.1 – La première colonne reprend les différentes techniques que nous avons envisagées dans notre état de l'art. La deuxième correspond à la contrainte selon laquelle nous ne voulons faire aucune hypothèse sur l'environnement de la cible. Les colonnes trois à cinq détaillent différents types de mouvements relatifs de la cible par rapport à la caméra, (respectivement les translations, les rotations dans le plan (p), et les rotations hors du plan (hp)). La sixième colonne nous indique la robustesse des techniques face au changement des conditions d'éclairage. Enfin, la dernière colonne nous indique si ces techniques sont applicables dans le cadre d'un suivi en temps réel. *KO (ech) indique que la technique échoue pour une translation, non pas dans le plan orthogonal à la caméra, mais bien lors d'un rapprochement ou d'un éloignement par rapport à la caméra.*

2.4.5 Discussion

Au vu de tout ceci, nous pouvons tenter de dégager l'approche la mieux adaptée a priori.

Parmi l'ensemble des méthodes de type distribution de couleurs, deux méthodes se démarquent. D'une part, le CamShift, qui est l'aboutissement des méthodes de rétro-projection, permet d'estimer la taille et l'orientation de la cible mais ne fournit pas de résultats satisfaisants pour des scènes de la vie réelle. D'autre part, l'intersection d'histogrammes, impose moins de contraintes sur l'environnement, mais ne dispose que d'une faible tolérance aux changements de taille et d'orientation.

Cependant, les travaux de nos prédécesseurs ont montré que l'intersection d'histogrammes est digne d'intérêt ; il est possible de lui adjoindre des traitements supplémentaires afin d'estimer la taille et de conférer à cette technique une robustesse relative aux changements d'illuminants.

Parmi l'ensemble des méthodes de type géométrique, les contours actifs sont sensibles à leur environnement proche, et se rapprochent, de par leurs possibilités et leurs faiblesses, de la méthode du CamShift. En effet, les points qui leur font défaut sont la robustesse à la luminosité et l'influence trop importante de l'environnement.

La méthode par corrélation croisée souffre des mêmes inconvénients que l'intersection d'histogrammes en ce qui concerne les rotations et les mises à l'échelle.Les paramètres de taille et d'orientation doivent être connus a priori, c'est-à-dire avant détection. Dans la corrélation croisée, la fenêtre de recherche et le modèle doivent être remis à l'échelle et orientés de façon **très précise**.

En revanche, les histogrammes présentent, eux, un double avantage. Premièrement, leur nature statistique, faisant disparaître toute information spatiale, leur confère la propriété d'être indépendants des rotations (dans le plan). Même si la fenêtre de recherche doit être corrigée, le modèle ne doit pas être modifié. Deuxièmement, même si le modèle doit être remis à l'échelle avant localisation, la tolérance offerte par la mesure d'intersection permet de travailler avec seulement une **estimation** du facteur d'échelle. De plus, une fois le facteur d'échelle estimé, cette correction peut s'effectuer très rapidement à l'aide d'une simple multiplication scalaire.

Le hachage géométrique est la méthode qui semble correspondre de la manière la plus adéquate au cadre hypothétique que s'est fixé le MIPS. Son seul inconvénient réside dans la complexité des algorithmes y étant mis en oeuvre pour la taille, trop importante pour envisager une mise en application temps réel sur des machines actuelles. Cependant, étant donné que la complexité des algorithmes impliqués est polynomiale et de degré relativement faible, cette piste ne devrait pas être exclue des recherches futures puisque l'évolution technologique apportera tôt ou tard une solution à ce manque de puissance de calcul. D'autant plus que cette approche solutionne naturellement un certain nombre de problèmes relativement compliqués tels que la gestion des mises à l'échelle et des rotations.

Les deux méthodes que nous prônons, après cette brève analyse, sont d'une part le hachage géométrique, dont le seul handicap est la complexité, et les méthodes d'intersection d'histogrammes d'autre part.

La généralisation des caméras couleurs, de même que l'intérêt porté, à la fois par la littérature et par nos prédécesseurs, aux méthodes d'intersection d'histogrammes nous incitent à nous concentrer sur cette méthode. Par ailleurs, il semble plus aisé de reconnaître un objet par ses couleurs que par sa forme, argument qui met lui aussi en avant la méthode d'intersection d'histogrammes. L'utilisation de la couleur dans le suivi d'objets est une voie qui a peu été explorée dans la littérature, qui ne permettra pas de résoudre tous les problèmes, mais qui mérite d'être développée.

2.5 Conclusion

Dans ce chapitre, nous avons commencé par expliquer la formation des images couleurs, concept essentiel pour la suite de notre travail. Nous avons ensuite établi un état de l'art, n'ayant pas la prétention d'être exhaustif, de différentes techniques envisageables dans le cadre d'un suivi d'objets.

Afin de déterminer quelle technique nous allons utiliser, nous avons comparé ces différentes techniques sur base du cadre d'hypothèses et de contraintes que s'est fixé le MIPS. Aucune des techniques envisagées ne nous apporte entière satisfaction, mais deux méthodes se démarquent malgré tout. D'une part la méthode du *CHS*, utilisant l'information couleur des objets et d'autre part le hachage géométrique utilisant, quant à lui, les particularités géométriques des objets.

Nous nous sommes tournés vers la méthode d'intersection d'histogrammes, car il semble plus aisé de retrouver un objet par ses couleurs que par sa forme. Par la suite, nous utiliserons la méthode d'intersection d'histogrammes couleurs, nous étudierons ses limites et nous envisagerons des solutions pour étendre le champ d'application de *CHS*. Un des enjeux sous-jacent à cette étude sera de mettre en évidence le potentiel de la couleur dans la localisation d'objets en temps réel.

Chapitre 3

Algorithme *CHS*

3.1 Introduction

Dans ce chapitre, nous présentons la méthode du ***CHS*** (**C**olour **H**istogram **S**imilarity) introduite par Buessler [8, 7] permettant de localiser, dans une image une couleur, une cible connue représentée par un histogramme.

Dans une première section, nous introduisons le concept d'histogramme couleur, central à la méthode du *CHS*, et permettant de représenter les images de façon compacte.

La deuxième section est dédiée à la présentation de l'algorithme de base du *CHS*. Dans la troisième section, nous discutons d'une part des optimisations qui le rendent utilisable en temps réel, et d'autre part de ses limitations, afin de définir les aspects qu'il conviendrait de travailler pour étendre son potentiel.

Ceux-ci nous mènent ensuite à définir plus précisément les différents concepts utilisés dans la méthode. Ainsi, les sections suivantes sont consacrées à l'étude des espaces couleurs et de leur quantification sur lesquels reposent les traitements, à l'optimisation de la signature couleur et enfin à la discussion de la mesure de similarité.

Les problèmes plus complexes mis en évidence par l'analyse des limitations sont étudiés plus largement dans des chapitres ultérieurs.

3.2 Modélisation de la cible

L'utilisation d'images couleurs dans le cadre de la localisation en temps réel ne saurait être envisagée sans compression préalable de l'information.

On constate que, dans les images délivrées par les caméras numériques, un très grand nombre (la quasi-totalité) des pixels sont de couleurs différentes. Un premier traitement de quantification[1] et d'indexation[2] visant à réduire cette diversité sont

[1] La quantification permet de réduire la diversité des couleurs dans un espace couleur.

[2] L'indexation est un processus visant à encoder une image avec un nombre de couleur plus limité. Le triplet de variables colorimétriques de chaque pixel est remplacé par un indice pointant dans une

effectués. Un paramétrage correct de l'algorithme de quantification permet de diminuer drastiquement la quantité d'information tout en gardant une bonne description de l'image d'origine (figure 3.14).

Une comparaison pixel par pixel des images de référence et de la scène serait beaucoup trop sensible aux variations de taille, d'orientation ainsi qu'aux fluctuation du signal couleur[3] ou de la luminosité.

Nous cherchons donc une signature d'image (qui est le résultat d'une fonction de compression[4] appliquée à l'image) qui soit invariante ou peu sensible à ces variations. De nombreuses signatures ont déjà été imaginées [30]. Swain et Ballard [29] ont montré que l'histogramme couleur était une signature intéressante, simple, compacte, facile à calculer et relativement insensible aux variations de pose. Un travail reste cependant à faire au niveau de la dépendance à la luminosité.

Dans la méthode du *CHS*, la mesure de similarité utilisée est la mesure d'intersection de Swain et Ballard. Le MIPS a étudié une utilisation directe de l'histogramme pour une localisation dans l'image par formation d'une surface de similarité (cfr. :3.3.2).

L'histogramme couleur représente la distribution des couleurs dans l'image. Formellement, si l'image est indexée avec une quantification réduisant le nombre de couleurs à P classes de couleurs $C_1, \ldots, C_P$, dont les couleurs représentantes sont $c_1, \ldots, c_P$, alors l'histogramme H_{im} de l'image im est un vecteur à P composantes, noté $(H(1), \ldots, H(P))$. Chaque $H(i)$ comptabilise le nombre de pixels dans l'image indexée ayant pour couleur c_i. La somme sur les $H(i)$ égale la taille de l'image :

$$\sum_{i=1}^{N} H(i) = m.n \tag{3.1}$$

où m et n sont respectivement la largeur et la hauteur de l'image im.

Par ce choix, on fait l'hypothèse que la distribution de couleur de l'objet cible l'identifie dans l'environnement dans lequel elle va évoluer. Si l'objet est suffisamment riche en couleur, cette hypothèse ne devrait pas se révéler trop contraignante et pourrait rentrer dans le cadre de la première hypothèse que nous avons formulée lors de la définition de la problématique (hypothèse cible / environnement).

3.3 Présentation

La localisation d'un objet cible dans un environnement peut se scinder en deux grandes étapes. La première est une étape préalable dans laquelle le modèle de suivi

palette de couleurs.

[3] En effet, même lors de l'observation d'une scène fixe (sans aucune variations quelconques de luminosité, ...), le signal couleur rendu par la caméra n'est pas constant (bruit, défaillance des certaines capteurs, ...). Ce phénomène est d'autant plus marqué que le capteur est de faible qualité.

[4] Le type de compression est souvent destructif, c'est-à-dire qu'il dégrade l'information ou en omet une partie.

(un histogramme pour le *CHS*) de la cible est calculé. La seconde consiste en son suivi proprement dit, tout au long d'une séquence d'images. Il est donc répété pour chaque nouvelle image.

3.3.1 Acquisition de la cible et initialisation

La toute première opération consiste en l'acquisition de l'image de référence de l'objet cible. Dans les implémentations expérimentales que nous avons mises en oeuvre, cette opération s'effectue par sélection d'une zone rectangulaire dans l'image scène.

Cette image cible est ensuite indexée afin de réduire son nombre de couleurs et son histogramme H_{ref} est finalement calculé. Celui-ci constitue le modèle de suivi qui sera utilisé au long de la séquence pour représenter l'objet cible.

3.3.2 Processus de suivi

Une fois le modèle constitué, il est ensuite comparé aux signatures de toutes les sous-images de même taille que les images de la séquence.

Acquisition et indexation

La première opération d'une itération du *CHS* consiste à acquérir une image scène dans laquelle l'objet cible doit être localisé. Celle-ci est ensuite indexée avec les mêmes paramètres que ceux utilisés pour indexer l'image cible.

Construction de la surface de similarité

La deuxième phase consiste à mesurer à quel point les différentes sous-images de la scène (de mêmes tailles que l'image cible) ressemblent à l'image cible. Supposons que celle-ci ait une taille de $m.n$ pixels.

L'image scène, de taille $M.N$ est balayée séquentiellement avec une fenêtre de recherche rectangulaire $FR(x,y)$ de $m.n$ pixels. Pour les différentes positions (x, y) de celle-ci, la valeur de la mesure de similarité de Swain et Ballard [29] entre son histogramme $H_{FR(x,y)}$ et l'histogramme de référence H_{ref} est calculée[5] :

$$s(x,y) = d_{\cap}(H_{ref}, H_{FR(x,y)}) = \frac{\sum_{i=1}^{N} min(H_{ref}\,(i), H_{FR(x,y)}(i))}{\sum_{i=1}^{N} H_{ref}(i)} \tag{3.2}$$

Elle est ensuite reportée sur une surface, que nous appellerons dans la suite surface de similarité. Chaque point de coordonnée (x, y) de cette surface de taille $(M - m + 1).(N - n + 1)$ représente donc la valeur de similarité entre l'histogramme de la fenêtre de coin supérieur gauche de coordonnée (x, y) et l'histogramme de l'image cible.

Cette surface de similarité peut être représentée de différentes façons, soit en niveau de gris, soit à l'aide d'une gradation de couleur ou encore en trois dimensions. En raison des deux premiers modes de représentations, nous appellerons parfois le couple (x, y) pixel de la surface de similarité.

[5] D'autre mesure de similarité ont été proposée : cfr 3.7 .

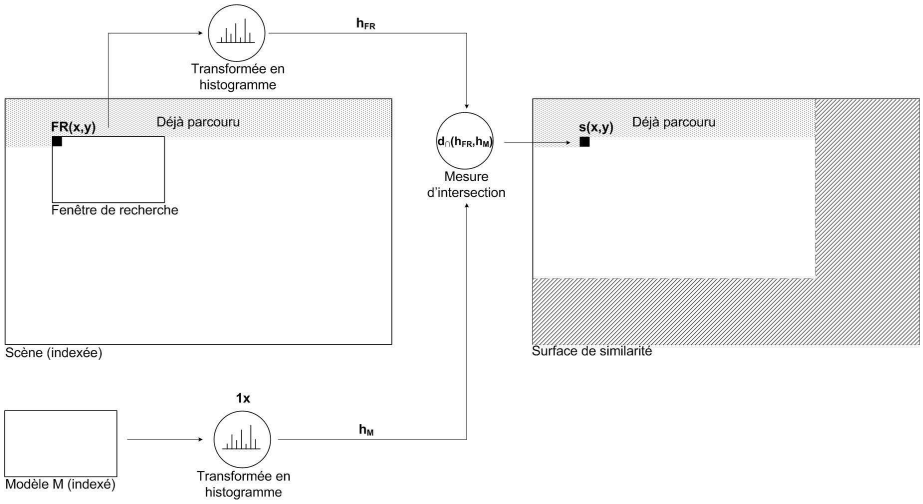


FIG. 3.1 – Calcul de la surface de similarité

Analyse de la surface et détermination de la position

Les coordonnées du coin supérieur gauche de la cible (x^*, y^*) sont celles qui sont telles que la valeur de la surface de similarité en ce point soit maximale, idéalement un.

$$(x^*, y^*) = argmax_{(x,y)} s(x, y) \tag{3.3}$$

Cette solution correspond à la position la plus probable de la cible.

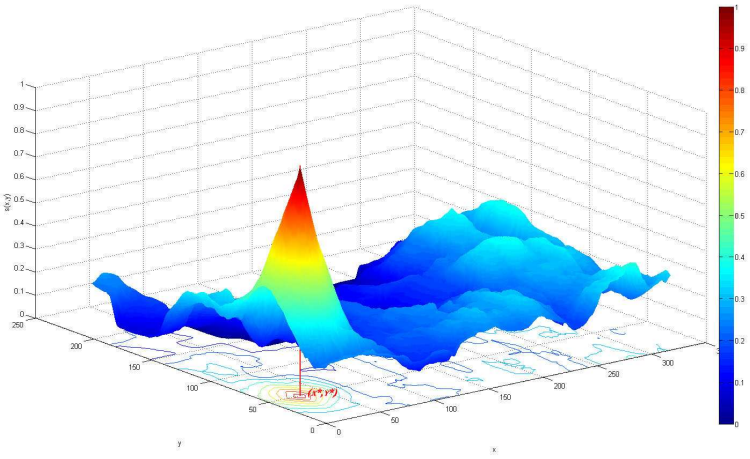
3.4 Mise en oeuvre et limitations

A notre arrivée à Mulhouse, une implémentation optimisée de l'algorithme de base du *CHS* était déjà existante. Dans la première section, nous présentons brièvement deux principales optimisations qu'a réalisées le MIPS en vue de la rendre utilisable en temps réel (Différentiel et SIMD). Nous discutons également de deux optimisations supplémentaires que nous avons proposées afin de tirer parti de possibilités matérielles en vue d'accélérer l'ensemble des traitements en jeu dans la méthode du *CHS*.

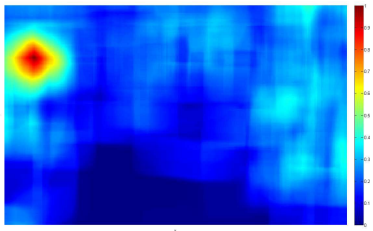
L'algorithme de base du *CHS*, ainsi que les améliorations qui ont été apportées dans [1] souffrent d'un certain nombre de limitations. Nous mettons celles-ci en évidence dans une deuxième section, ce qui permettra de définir les aspects à travailler par la suite.

Celles-ci nous ont conduits à une analyse plus détaillée des différents concepts sur lesquels repose l'algorithme.

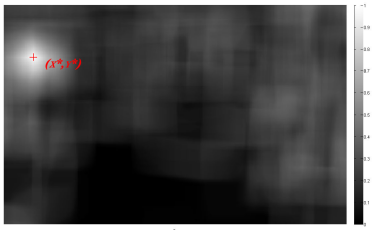
Dans une troisième section, nous nous intéresserons au choix de l'espace couleur qui supporte les traitements ainsi qu'à sa quantification, ces deux paramètres ayant des impacts importants sur les résultats.



(a) Surface 3D



(b) Colormap(jet)



(c) Colormap(gray)

FIG. 3.2 – Différentes représentations de la surface de similarité.

La section quatre nous permettra d'analyser plus finement la signature couleur. Nous proposons une piste de recherche permettant d'optimiser sa construction.

Enfin, la dernière section sera réservée à l'étude de la mesure de similarité et à l'analyse de la surface de similarité.

3.4.1 Optimisation et performances

Malgré la simplicité des opérations en jeu dans le processus de calcul de la surface de similarité, celle-ci s'avérerait trop lourde à calculer pour être utilisée dans un cadre temps réel sans aucune optimisation.

Nous présentons ici différentes optimisations permettant d'accélérer le calcul de la surface de similarité, certaines tirant parti de ses propriétés et d'autres exploitant les possibilités matérielles disponibles (jeux d'instruction SIMD, architecture parallèle, ...).

Différentiels

La première optimisation qu'a réalisée le MIPS vient du constat qu'entre deux positions successives de la fenêtre de recherche seuls quelques pixels varient. La position

relative des pixels n'étant pas importante dans le cas des histogrammes, il suffit, pour obtenir l'histogramme de la position suivante, d'ajouter la contribution des pixels qui appartiendront à cette fenêtre de recherche, mais qui n'appartiennent pas à celle que l'on vient de calculer, et de soustraire la contribution de ceux qui appartenaient à la fenêtre de recherche précédente mais plus à celle-ci.

Afin de calculer la surface de similarité, nous faisons serpenter la fenêtre de recherche sur l'image. Pour un déplacement horizontal de la fenêtre de recherche, la contribution de chacun des pixels de la première colonne de la fenêtre de recherche précédente doit être retranchée et celle des pixels de la dernière colonne de la fenêtre de recherche actuelle doit être rajoutée. De manière similaire, pour un déplacement horizontal nous retrancherons la contribution de la première ligne et ajouterons la contribution de la dernière colonne. Il s'ensuit que seul le premier histogramme doit être calculé de manière traditionnelle, tous les autres histogrammes en sont successivement dérivés.

SIMD

La seconde amélioration apportée par la groupe MIPS est, quant à elle, propre aux processeurs Pentium 4. L'utilisation des instructions SSE2 nous permet de paralléliser le calcul de plusieurs fenêtres de recherche. Si, selon nos estimations, les instructions SSE prennent en moyenne 2 fois plus de temps qu'une instruction normale, elles nous permettent cependant de calculer 8 fenêtres de recherches simultanément, et donc d'augmenter par 4 les performances obtenues.

En utilisant un mot de 16 bits pour chacun des bins de l'histogramme, il s'ensuit qu'un registre de 128 bits pourra contenir soit 8 bins différents, soit ce même " bin " pour 8 histogrammes différents. Cette dernière approche permet à l'algorithme du groupe MIPS de calculer simultanément 8 histogrammes, comme le montre le schéma sur la figure [3.3]. La méthode utilisant le SSE2 implémentée par le groupe MIPS utilise une fenêtre de recherche ayant 7 pixels de hauteur en plus que celle de la méthode classique, afin de contenir tous les pixels intervenant dans les 8 fenêtres de recherche normales pour lesquelles les histogrammes sont calculés simultanément. Cependant, la première ligne de cette fenêtre de recherche contribue au premier histogramme, mais pas au suivant et ainsi de suite. Il nous faut donc utiliser des masques afin de déterminer, en fonction de la ligne où nous nous trouvons, à quels histogrammes ces pixels contribuent.

Pour le reste, l'algorithme utilisant les instructions SSE2 se déroule de manière similaire à la méthode différentielle. Le décalage d'une position à l'autre sur une même ligne s'effectue comme dans la méthode précédente, si ce n'est que nous utilisons les masques pour ajouter et retrancher la contribution des pixels aux huit histogrammes que nous calculons actuellement. En ce qui concerne les décalages horizontaux, nous décalons de 8 lignes en une fois, car les fenêtres de recherche des 8 précédentes ont étés calculées simultanément au moyen des instructions SSE2.

Une étude évaluant l'apport de l'utilisation conjointe de ces deux dernières opti-

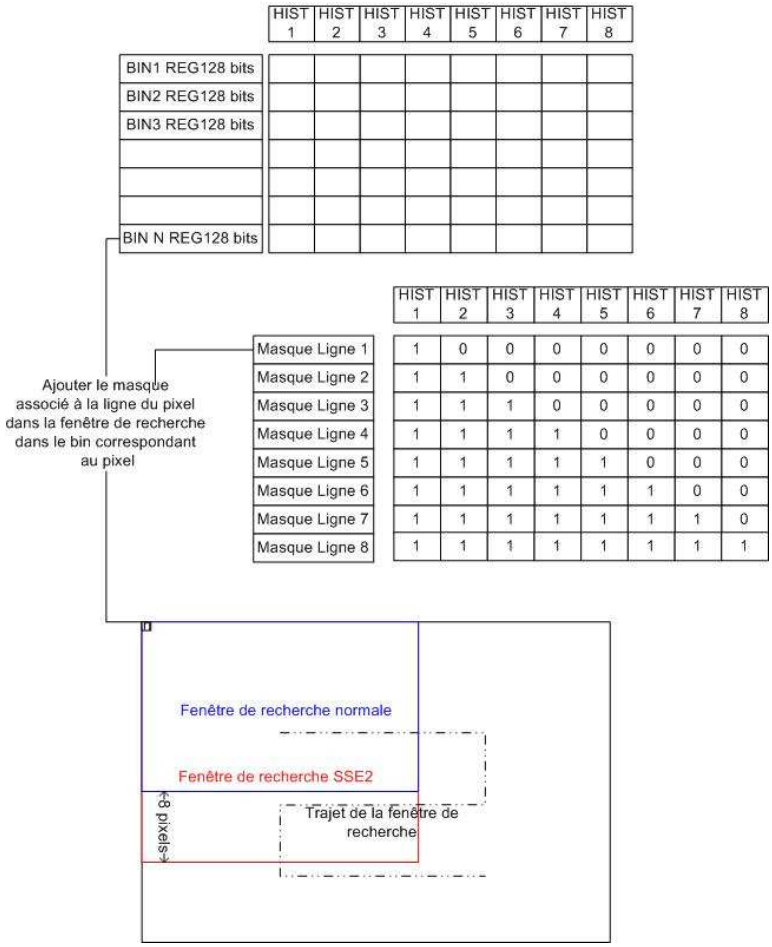


FIG. 3.3 – Calcul de la surface de similarité via les optimisations SIMD.

misations est disponible en annexe C.

Multi threading

Il pourrait également être intéressant de profiter des architectures parallèles, telles les plates-formes multiprocesseurs ou encore les processeurs "dual core" de plus en en plus répandus sur le marché et dont le prix est maintenant très abordable. Le MIPS dispose d'ailleurs de machines biprocesseurs Intel Xeon 2.4GHz.

Le calcul de la surface de similarité, même optimisé comme présenté ci-dessus, pourrait être partitionné et distribué sur plusieurs processeurs. La détermination de la solution finale se ferait très facilement par recherche du maximum sur l'union des éléments de la partition.

Une approche simple, permettant de multi-threader le calcul sans modifier l'implémentation existante du *CHS*, consisterait à effectuer une découpe horizontale ou verticale et à distribuer le travail à des threads exécutant différentes instances du code et qui calculent l'intersection d'histogrammes. Si T est le nombre disponible de processeurs, il devrait en résulter une division du temps de calcul par un facteur lé-

gèrement inférieur à T, ceci étant dû au fait que la découpe en sous-surfaces empêche de profiter pleinement du calcul "par différence" des histogrammes.

Si $W_i, i = 1, \ldots, T$ est une partition d'une image I et

$$Z^* = \left\{(x_i^*, y_i^*) = argmax_{(x,y)\in W_i} s(x,y) \quad i = 1, \ldots T\right\}$$

l'ensemble des T solutions du *CHS* pour les T sous-images, alors la solution du *CHS* pour l'image I est donnée par

$$(x^*, y^*) = argmax_{(x,y)\in Z^*} s(x,y)$$

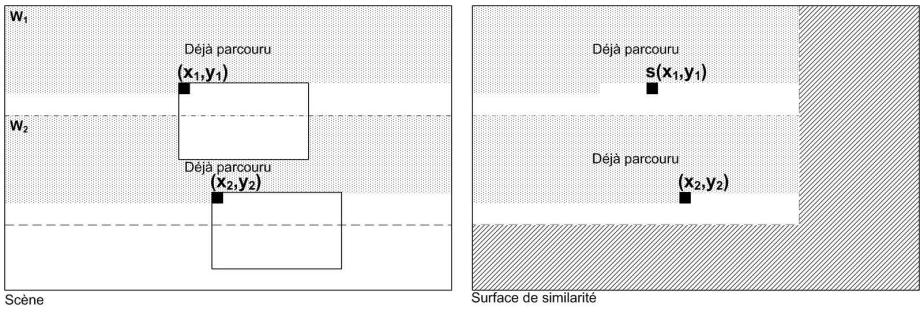


FIG. 3.4 – Multithreading (T=2) du calcul de la surface de similarité.

Fenêtrage

Une possibilité supplémentaire pour accélérer les traitements serait de fenêtrer[6] les images délivrées par la caméra pour n'en garder que des zones où l'on est sûr que la cible se trouve. Ainsi, en fonction du stade auquel est effectué le fenêtrage, il est possible de réduire plus ou moins fortement le temps d'exécution d'une itération.
La caméra industrielle que nous avons utilisée durant notre stage nous permettait de fenêtrer les images matériellement, c'est-à-dire dès l'acquisition. Nous pouvions donc enregistrer un gain de temps sur la numérisation, sur la transmission vers la mémoire de l'ordinateur, sur la conversion de format, sur l'indexation, sur le calcul de la surface de similarité ainsi que sur la détermination de son maximum.

3.4.2 Evaluation et limitations

Les travaux de Buessler et al. [7, 8, 9], d'André et Frippiat [1] confirment largement l'intérêt de développer la méthode du *CHS*. Cependant, l'algorithme de base ainsi que les améliorations qui y ont été apportées souffrent encore de certaines limitations que nous mettons ici en évidence, ce qui nous permet par la même occasion de définir les aspects de l'algorithme à travailler et les directions dans lesquelles nous allons orienter le reste de notre étude.

[6] Ne considérer que la sélection d'une sous-image (ou fenêtre) de l'image originale

Potentiel descriptif de la signature et paramétrage

Même si les différents travaux précités montrent que le potentiel descriptif[7] des histogrammes est important, celui-ci est d'une part limité et d'autre part assez complexe à paramétrer de façon optimale.

Cas limites Il est évident que deux objets différents, mais présentant une même distribution de couleur ne peuvent pas êtres distingués par une technique de traitement d'image telle que le *CHS*. D'autre part, une même distribution de couleur peut être la caractéristique de deux images complètement différentes 3.5, la fonction transformant une image en histogramme n'étant pas injective. Afin d'outrepasser cette dernière limitation, des solutions utilisant le corrélogramme couleur, incorporant l'information de texture[8] [1, 19] ont été envisagées. Il a cependant été montré que celles-ci, en plus d'être coûteuses, ne fournissaient que de modestes résultats. Nous ne nous y sommes donc pas attardés.

(a) Impossibilité de distinguer les carrés rouges

(b) Impossibilité de distinguer les deux drapeaux

FIG. 3.5 – Des images de même distributions de couleur ne peuvent être distinguée par le *CHS*

Il est également intéressant de remarquer que l'utilisation de la couleur ne permet pas de suivre n'importe quelle cible dans un environnement donné. Il n'y a pas forcément de solution algorithmique et il faudra dès lors vérifier que l'objet présente des couleurs suffisamment discriminantes pour assurer son suivi de façon fiable (et éventuellement réaliser un marquage spécifique, bien que ceci nous éloigne de notre cadre de contraintes).

Paramétrage L'histogramme couleur est déterminé par deux paramètres. Le premier (implicite) est l'espace couleur dans lequel sont représentées les images. Le second est la quantification déterminant les classes de l'histogramme.

Si le choix de l'espace sur lequel repose l'histogramme n'a que peu d'incidence sur son potentiel descriptif (quelque soit l'espace choisi, on peut toujours trouver une quantification fournissant le même potentiel descriptif, en mettant les points couleurs des classes en correspondance), la quantification qui lui est associée fait en revanche l'objet d'un compromis entre d'une part capacité à décrire un objet de façon plus ou moins discriminante et d'autre part une certaine robustesse (ou tolérance) aux variations du signal couleur (bruit, imprécision des capteurs, changements de luminosité, ...).

[7] Le potentiel descriptif d'une signature caractérise sa capacité à discriminer l'image qu'elle décrit parmi d'autres.

[8] La texture caractérise la disposition relative des pixels au sein d'une image.

Dans la section 3.5.1, nous définissons plus formellement le concept d'espace couleur pour ensuite discuter en 3.5.2 des différentes méthodes de quantification que nous avons envisagées. Celles-ci permettent la construction de quantifications plus flexibles et plus précises que celles envisagées par nos prédécesseurs dans [1], limitées par des contraintes d'allocation mémoires. De plus, les méthodes proposées permettent de ne travailler que sur les parties réellement utilisées de l'espace couleur et non sur l'espace entier, supprimant ainsi le volume d'information inutile.

Ces méthodes à disposition, nous avons ensuite pu nous attaquer au problème de l'optimisation de la signature couleur, et plus particulièrement au nombre de classes de celle-ci (3.6.3). Celle-ci est basée sur l'optimisation d'un critère statistique proposé par Birgé et Rozenholc [3] balançant fidélité de description de l'échantillon et nombre de classes de l'histogramme. Encore en développement, elle ne fournit que de modestes résultats, mais nous proposons d'explorer cette piste plus profondément.

Enfin, nous avons construit et implémenté notre propre algorithme d'indexation rapide (3.5.3) basé sur ces méthodes et indexant les images en temps linéaires par rapport à leur taille. Son efficacité en temps réel à été validée lors de nombreuses expériences, le traitement d'une image étant de l'ordre de la milliseconde sur notre machine de test.

Condition sur le spectre des sources Le potentiel descriptif d'un histogramme couleur est également limité par une contrainte d'ordre physique. Le spectre réfléchi par une surface dépend de la source à l'origine du rayonnement et qui permet de la percevoir. Le spectre de la source doit donc être suffisamment riche que pour permettre à la surface de réfléchir un spectre suffisamment complet que pour la démarquer des autres. Idéalement, la source devrait émettre une lumière blanche, composée de toutes les longueurs d'ondes.

Dans un cas extrême, une cible orange vue sous une lumière rouge, serait perçue comme rouge, la composante jaune inexistante dans le spectre de la source ne pouvant être réfléchie (figure 3.6).

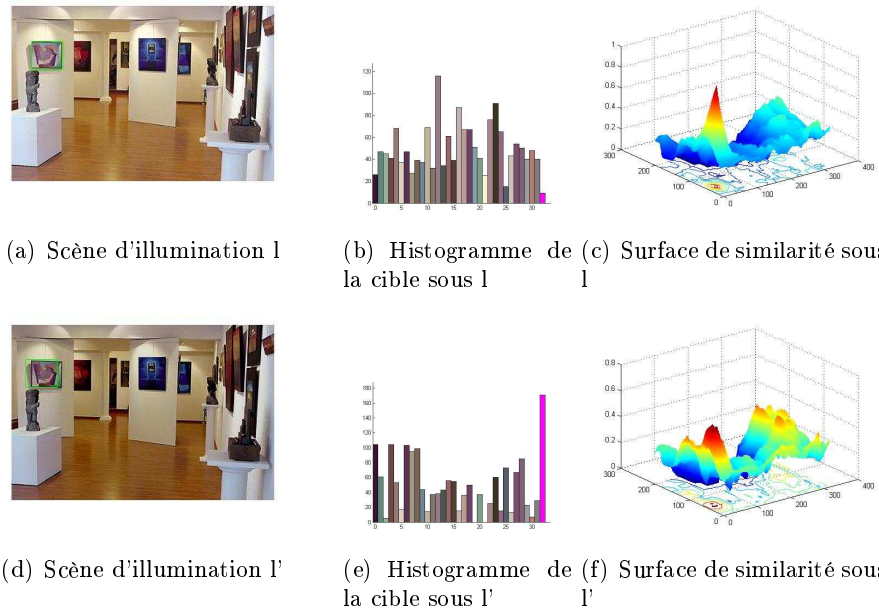
(a) Spectre de la source assez riche

(b) Spectre de la source trop pauvre

FIG. 3.6 – Signal couleur et richesse du spectre de la source

Sensibilité aux variations d'illumination

L'algorithme de base du *CHS* est sensible aux variations d'illuminations. Spatiales ou spectrales, celles-ci modifient le spectre réfléchi par l'objet, avec pour conséquence, une variation de sa signature couleur. La comparaison de celle-ci au modèle de suivi perd donc tout son sens étant donné qu'il représente l'objet vu sous une illumination différente, faisant très souvent échouer la localisation. La figure 3.7 illustre l'impact du changement de luminosité (créé artificiellement) sur la signature couleur et sur la surface de similarité)



(a) Scène d'illumination l (b) Histogramme de la cible sous l (c) Surface de similarité sous l

(d) Scène d'illumination l' (e) Histogramme de la cible sous l' (f) Surface de similarité sous l'

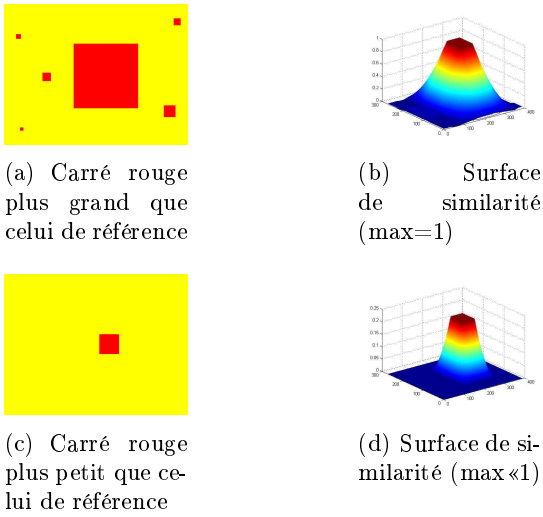
FIG. 3.7 – Recherche d'une réference (tableau) capturée sous une illumination l, dans deux scènes d'illumination l (haut) et l' (bas).

En vue d'une application en conditions réelles du *CHS* , il conviendrait de l'affranchir des variations de conditions d'illumination ou, du moins, d'y apporter une certaine robustesse. Par la suite nous nuancerons entre d'une part de très faibles variations, auquel cas nous parlerons de tolérance, et d'autre part de fortes variations, cas dans lequel nous parlerons plutôt d'indépendance aux conditions d'illumination.

Le chapitre 4 est entièrement consacré à l'étude de cet épineux problème. Nous commençons par faire état de l'ensemble des pistes possibles et déjà explorées pour choisir certaines directions. De façon résumée, nous avons d'une part affiner l'étude de la réponse apportée par la quantification de l'espace couleur en terme de tolérance, et d'autre part exploré une autre piste, reposant sur les travaux de Finlayson, et dans laquelle un prétraitement des images est effectué en vue d'annuler les effets des variations de luminosité. Ces différentes pistes de solutions ont été évaluées sur des jeux de test à l'aide d'un critère que nous avons développé (4.2) et permettant de mesurer leurs impacts sur la qualité de la solution du *CHS*.

Variation de taille apparente

Les variations de distance entre l'objet cible et la caméra induisent dans la séquence d'images une variation de taille apparente de l'image cible. La fenêtre de recherche (rectangulaire dans la version originale) doit donc être - au moins approximativement - remise à l'échelle en vue de construire son histogramme pour le comparer au modèle de suivi. Même si l'intersection d'histogrammes offre une certaine tolérance à de faibles variations de taille apparente, une localisation précise (figure) et fiable (figure) ne peut être assurée qu'en connaissance de la taille exacte.



(a) Carré rouge plus grand que celui de référence

(b) Surface de similarité (max=1)

(c) Carré rouge plus petit que celui de référence

(d) Surface de similarité (max «1)

FIG. 3.8 – Recherche d'une réference (tableau) capturée sous une illumination l, dans deux scènes d'illumination l (gauche) et l' (droite)

Le chapitre 5 est entièrement dédié à l'étude de ce problème relativement complexe. Nous avons principalement approfondi la technique d'estimation de la taille du *CHF* 5.4 initiée par Buessler et Urban dans [9] comme proposé dans les perspectives de [1]. Celle-ci donnant de bons résultats mais souffrant toutefois d'un paramétrage critique, nous avons proposé une solution originale et radicalement différente basée sur l'analyse d'une famille d'indicateurs. Les résultats sont prometteurs, mais il conviendrait de continuer son développement.

Qualité de la solution

Comme l'ont mis en évidence nos prédécesseurs, la nature de la solution du *CHS* (maximum de la surface de similarité), est telle qu'elle existe toujours. Représentant la position la plus probable (au sens de la mesure de similarité utilisée), elle peut parfois correspondre à une région qui n'est pas celle où se trouve réellement la cible, ce qui peut arriver lorsque

- des changements d'illuminations viennent rendre une autre région de la scène plus semblable au modèle de suivi que la cible,
- une mauvaise estimation de la taille ne permet pas de placer correctement la fenêtre de recherche sur la région de la cible,
- la cible est partiellement masquée ou ne se trouve pas dans la scène,
- ...

Une combinaison d'indicateurs permettant d'accepter ou de rejeter la solution fournie par le *CHS* a été développée dans [1]. Bien que leur nécessité soit indéniable, nous ne nous sommes penché que sommairement sur la question, faute de temps. Cette piste ne devrait toutefois pas être exclues des recherches futures.

Forme et déformation

L'implémentation actuelle du *CHS* est limitée à la localisation de cibles rectangulaires. Si celles-ci ne sont pas trop allongées, le paramètre d'orientation peut dans un premier temps être négligé. La généralisation de l'algorithme au cas des formes quelconques pourrait s'effectuer à l'aide d'un masque, le paramètre d'orientation devenant alors plus critique (figure 3.9). L'algorithme doit donc en plus considérer une certaine plage d'orientations possibles, ce qui alourdit considérablement le traitement.

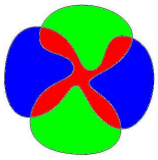


FIG. 3.9 – Pour une cible de forme complexe, le paramètre d'orientation prend un rôle critique. Vert = cible, Bleu = masque, Rouge = intersection entre le masque et la cible

Dans le chapitre 6, nous discutons d'une technique que nous avons imaginée et qui utilise le *CHS* de manière plus locale (recherche de motifs). Celle-ci permet d'une part d'effectuer le suivi de cible de forme quelconque, et d'autre part d'estimer taille et orientation simultanément.

Enfin, la dernière problématique que nous avons identifiée concerne la gestion des variations de perspectives par l'algorithme du *CHS*.

D'une part, celles-ci peuvent induire des déformations de la projection de l'objet cible que l'on recherche. Le balayage avec une fenêtre rigide ne permet pas de prendre en compte ces variations. Il conviendrait de s'interroger quant aux impacts de celles-ci sur l'algorithme du *CHS* (et sur la surface de similarité en particulier). Gagnerait-on réellement à introduire un nouveau paramètre de déformation de la fenêtre de recherche ? D'autre part, ces variations de perspectives peuvent faire apparaître dans les images de la séquence la projection de portions de surfaces initialement cachée lors de la capture de l'image de référence.

Les variations de perspectives et de pose de l'objet soulèvent une série de questions passionnantes pouvant probablement faire l'objet d'un travail spécifique. Pour le moment, nous considérons que le principe du *CHS* permet une bonne tolérance à des variations (limitées) de perspectives.

3.5 Espace couleur et indexation

3.5.1 Espace couleur

Introduction

Dans l'application qui nous concerne, nous devons choisir un espace de représentation des couleurs pour effectuer la quantification. Les résultats de la comparaison d'histogrammes dépendent de ce choix préalable.

Nous rappelons d'abord brièvement la sélection des meilleurs espaces potentiels (parmi les espaces classiques) qui a été effectuée par nos prédécesseurs, pour ensuite présenter trois espaces couleurs que nous utilisons par la suite, à savoir *RGB* qui est l'espace natif de nombreux périphériques d'acquisition et largement utilisé en imagerie numérique, *HSV* qui est un système séparant l'information de luminance et chrominance, et enfin LST, un espace plus original visant à amener une réponse à des problèmes de cohérence dans *HSV*, notamment au niveau de la notion de norme.

Dans leur mémoire, André et Frippiat ont réalisé une étude comparative des différents espaces visant à sélectionner le(s) meilleur(s) espace(s) sur le(s)quel(s) le *CHS* offre les meilleures performances. Ils ont à cette fin réalisé des tests sur des ensembles d'images reprenant différents cas de figure (illuminations différentes, cibles fortement ou faiblement nuancées par rapport au fond, ...). Leur critère d'évaluation est une amélioration des performances du *CHS* par rapport au *RGB* standard, format natif de la caméra et pour lequel aucune conversion, coûteuse en temps de calcul, ne doit être effectuée avant traitement.

Ceux-ci proposent de retenir les espaces *RGB* et *HSV* fournissant des résultats équivalents, avec toutefois un léger avantage pour *HSV* en ce qui concerne la tolérance à des faibles variations de luminosités. Nous reviendrons sur cette réflexion dans le chapitre 4.

RGB

L'espace *RGB* (Red, Green, Blue ou RVB pour Rouge, Vert, Bleu) est un espace tridimensionnel de représentation de la couleur où celle-ci est décrite suivant trois quantités scalaires de rouge, de vert et de bleu, reportées le long de trois axes orthogonaux, formant la base du cube *RGB*. La couleur d'un point de cet espace est obtenue par synthèse additive des valeurs des trois composantes primaires.

Ce modèle est au centre de la colorimétrie numérique, non pas parce qu'il apporte des avantages particuliers par rapport à d'autres modèles mais simplement parce qu'il dérive de la technologie la plus souvent employée dans l'environnement numérique.

Les moniteurs LCD comme les moniteurs CRT sont basés sur le principe additif de trois couleurs primaires, rouge, vert et bleu et à partir desquels il est possible de reconstruire la quasi-totalité de l'ensemble des couleurs (principe de tri-variance visuelle). Les scanners et les appareils photos ou les caméras numériques séparent

les composantes *RGB* des ondes lumineuses qu'ils reçoivent (à l'aide par exemple de filtre de Bayer) pour les quantifier indépendamment l'une de l'autre et rendre compte de l'information sous forme d'un triplet (R,G,B). L'oeil humain, à l'aide de ses trois types de cônes, analyse également les images dans un modèle de type *RGB*.

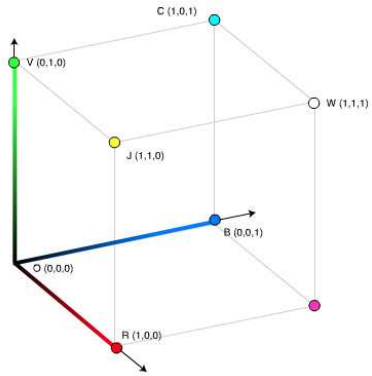


FIG. 3.10 – Représentation de l'espace couleur *RGB*, les trois axes orthogonaux portant les composantes R, G et B.

Remarquons aussi que ce mode de représentation par primaire rend *RGB* fortement corrélé, c'est-à-dire que certaines informations (telles la teinte ou la luminosité) sont communes à plusieurs composantes. Par exemple, une définition habituelle de luminosité dans *RGB* se définit par la relation :

$$l = \frac{(R + G + B)}{3} \tag{3.4}$$

Il existe d'autres définitions, mais celle-ci illustre parfaitement la corrélation qui lie les variables de *RGB*. Un traitement indépendant des composantes ne peut donc se faire sans une certaine perte d'information.

HSV

Introduit par Smith en 1978, *HSV* (Hue, Saturation, Value) est un espace dans lequel la couleur est repérée de façon géométrique et où elle est dissociée suivant des composantes plus intuitives pour l'être humain qu'un système de primaires tel *RGB*.

La composante H (Hue) représente la teinte ou tonalité chromatique, et est modélisée à l'aide d'un angle, variant de 0 ° à 360 ° .

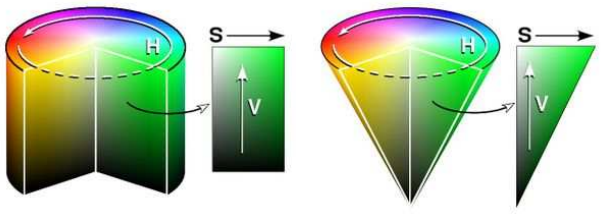
- H = 0 ° représente le rouge
- H = 60 ° représente le jaune
- H = 120 ° représente le vert
- H = 180 ° représente le cyan
- H = 240 ° représente le bleu
- H = 300 ° représente le magenta

La composante S quantifie la saturation d'une couleur. Intuitivement, ce paramètre mesure dans quelle proportion la couleur en question résulte d'un mélange entre une couleur pure et du blanc. Celle-ci varie de zéro à un et est reportée sur

une droite orthogonale à l'axe de symétrie du cône ou du cylindre, en fonction de la définition adoptée.
La composante V (Value) caractérise la luminosité qui, de façon intuitive, nuance entre "sombre" et "clair"'. Elle varie également entre 0 et 1 et est reportée le long de l'axe de symétrie du cône. L'axe V est généralement appelé "axe de gris".

Cette représentation polaire de la couleur possède deux singularités qui lui sont propres.

- Si la saturation est nulle (S=0), alors la teinte (H) est indéfinie. Lorsque S=0, les couleurs s'alignent le long de l'axe des gris, sur lequel la teinte n'y ayant aucun sens est bien sûr inexistante.
- Si la luminosité est nulle (V=0), alors la saturation (S) est indéfinie. En effet, lorsque V=0, la couleur correspondante est le noir pur, pour laquelle les notions de teinte et de saturation n'ont aucun sens. La représentation conique de *HSV* 3.11(a) permet de bien visualiser cet aspect.



(a) Définition cylindrique (b) Définition conique

FIG. 3.11 – Représentation des deux définitions de l'espace *HSV* (source wikipedia.org)

Les formules de conversion depuis l'espace *RGB* , pour la version cylindrique, sont données par :

$$H = \begin{cases} 60 \times \frac{G-B}{MAX-MIN} + 0 & si\ MAX = R\ et\ G \geq B \\ 60 \times \frac{G-B}{MAX-MIN} + 360 & si\ MAX = R\ et\ G < B \\ 60 \times \frac{B-R}{MAX-MIN} + 120 & si\ MAX = G \\ 60 \times \frac{R-G}{MAX-MIN} + 240 & si\ MAX = B \end{cases} \quad (3.5)$$

$$S = \frac{MAX - MIN}{MAX} \quad (3.6)$$

$$V = MAX \quad (3.7)$$

La version conique pouvant être obtenue en remplaçant la définition de saturation par la définition suivante :

$$S = MAX - MIN \quad (3.8)$$

LST L_1

Selon Jean Serra, les représentations polaires classiques de la couleur, telles que *HSV* présenté ci-dessus, conduisent à des mesures de brillance et de saturations aux

propriétés incohérentes. Dans [26], il propose de définir un nouvel espace, caractérisant la couleur en terme de teinte, saturation et luminosité mais en remplaçant ces deux dernières composantes par des normes[9] telles que L_1 ou MAX-MIN [10].

Serra a montré que la détection d'alignement de points dans des histogrammes bidimensionnels L/S permettrait d'isoler des zones d'ombre ou de reflet dans des images. C'est précisément cette hypothèse qui nous a amenés à nous intéresser à l'espace LST(L_1).

$$l = 3.m = r + g + b \tag{3.9}$$

$$s = \begin{cases} \frac{3}{2}(max - m) & si\ m \geq med \\ \frac{3}{2}(m - min) & si\ m \leq med \end{cases} \tag{3.10}$$

$$t = \frac{\pi}{3}\left[\lambda + \frac{1}{2} - (-1)^{\lambda}\frac{max + min - 2med}{2s}\right] \tag{3.11}$$

$$\lambda = \begin{cases} 0 & \text{si } r > g \geq b \\ 1 & \text{si } g \geq r > b \\ 2 & \text{si } g > b \geq r \\ 3 & \text{si } b \geq g > r \\ 4 & \text{si } b > r \geq g \\ 5 & \text{si } r \geq b > g \end{cases} \tag{3.12}$$

où, dans ces équations
- $m = \frac{1}{3}(r + g + b)$
- $min = min(r, g, b)$
- $max = max(r, g, b)$
- $med = mediane(r, g, b)$

Une implémentation efficace (en C, dans une librairie accessible depuis Matlab) de cette transformation a été réalisée afin de mettre à l'épreuve la théorie développée dans [26].
Nous avons en particulier voulu vérifier si cet espace permettait de mieux caractériser les objets, ou d'intégrer un prétraitement pour atténuer les effets d'ombres et de reflet. Toutefois, le fait que nous soyons parvenus à mettre en correspondance des alignements obtenus expérimentalement et les frontières délimitant l'espace L/S, nous porte à croire que ces alignements ne représentent en fait rien de plus que la saturation d'une ou plusieurs composantes R, G ou B (développements complémentaires disponibles dans l'annexe D).

3.5.2 Quantification et indexation

Afin de rendre fidèlement les images qu'elles acquièrent, les caméras numérisent très souvent vers des espaces couleurs finement discrétisés. Typiquement, la caméra que nous avons utilisée durant notre stage était configurée pour délivrer des images

[9] Mesure de distance
[10] (MAX = max{R,G,B}, MIN=min{R,G,B})

RGB en 24bpp (8 bits pour chaque composante R, G et B et donc 256 valeurs possibles), permettant ainsi de rendre compte de 16 millions de couleurs.

L'objectif est de reconnaître un objet en comparant ses couleurs avec celles d'un histogramme de référence alors que son image est perturbée par des variations de lumière, des ombres, des réflexions de couleur des objets proches ou encore par les fluctuations de la quantification effectuée par les capteurs de la caméras. Afin de fournir une certaine robustesse a toutes ces variations, la comparaison s'effectuera plus grossièrement, entre des classes de couleur.

Afin de réduire le nombre de couleurs possibles, deux opérations successives sont réalisées. L'espace couleur est d'abord quantifié (des ensembles de couleurs voisines sont assimilée à une couleur les représentant) et l'image est ensuite indexée sur base de la quantification déterminée (la couleur de chaque pixel est remplacée par celle à laquelle elle est assimilée).

Quantification

La première consiste en la création d'un ensemble de classes de couleur

$$C_i = C_i(c_i, f, r) \tag{3.13}$$

chacune déterminée par trois paramètres, et toutes associées, le cas échéant, à une même norme qui définit leur forme dans l'espace.

Le premier paramètre $c_i = (c_{i1}, c_{i2}, c_{i3})$, que nous appelons centre de classe ou noyau, est le représentant de la classe C_i, c'est-à-dire la couleur à laquelle seront assimilés tous les autres points couleurs de la classe et autour duquel elle sera construite.

Le deuxième paramètre $f = (f_1, f_2, f_3)$ est un vecteur de coefficients réels strictement positifs permettant de pondérer l'étalement de la classe suivant les trois dimensions de l'espace. Ceci permet de pouvoir donner une forme particulière aux classes, l'utilité de ceci se justifiant dans la section 4.3 traitant de la construction d'une quantification apportant une tolérance aux changements de luminosité.

Enfin, le troisième paramètre r appelé rayon de classe est un réel strictement positif déterminant la taille des classes. L'étendue de celles-ci sur chaque dimension étant donc fonction à la fois du coefficient de pondération associé et de ce dernier.

Nous avons envisagé trois structures de classe. La première est définie à l'aide d'inégalité sur des modules, composante par composante :

$$C_i^{uni} = C_i^{uni}(c_i, f, r) = \{(x_1, x_2, x_3) : |x_j - c_{ij}| \leq f_j.r \ , \ j = 1 \ldots 3\} \tag{3.14}$$

Les deux autres sont définies à l'aide des normes classiques L_1 et L_2 :

Index	Noyau	Classe représentée
1	c_1	C_1
⋮	⋮	⋮
P	c_P	C_P

TAB. 3.1 – Palette avec P classes.

$$C_i^1 = C_i^1(c_i, f, r) = \left\{ (x_1, x_2, x_3) : \sum_{j=1}^{3} \left| \frac{x_j - c_{ij}}{f_j} \right| \leq r \right\} \tag{3.15}$$

$$C_i^2 = C_i^2(c_i, f, r) = \left\{ (x_1, x_2, x_3) : \sqrt{\sum_{j=1}^{3} \left(\frac{x_j - c_{ij}}{f_j} \right)^2} \leq r \right\} \tag{3.16}$$

Dans un espace à base orthogonale (tel que RGB), ces classes prennent respectivement la forme de parallélépipèdes rectangles (figure 3.12), de tétraèdres et de sphères.

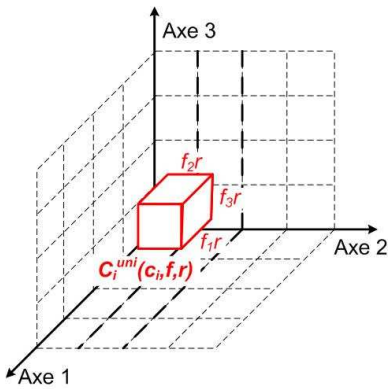
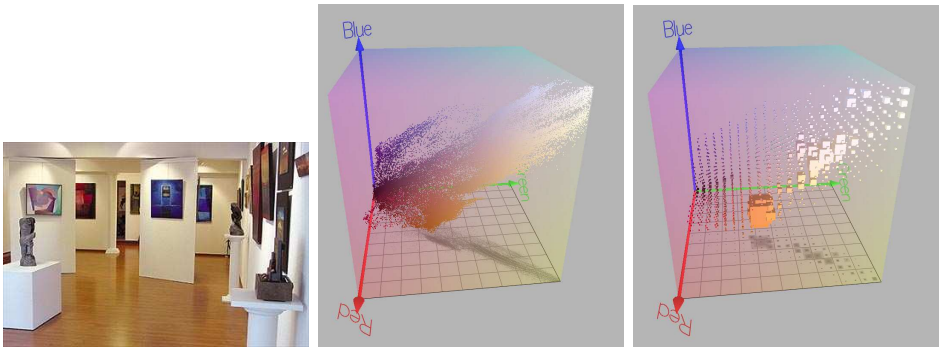


FIG. 3.12 – Représentation de la classe $C_i^{uni}(c_i, f, r)$ dans un espace à base orthogonale.

Techniquement, cette quantification est représentée par trois éléments logiciels :
- Une palette de couleur (ou map) qui est un vecteur associant les classes à leurs indices (qui seront utilisés pour remplacer les triplets de variables colorimétriques des pixels de l'image lors de l'étape de son indexation) (Tableau 3.1)
- Deux variables caractérisant la taille et l'étalement des classes suivant des directions privilégiées (fonction de la structure des classes)
- Une procédure de calcul représentant la norme ou l'ensemble d'inégalités définissant la structure des classes

On constate que l'ensemble des pixels d'une image n'occupe généralement pas l'intégralité de l'espace couleur, mais se regroupe plutôt dans des zones d'assez fortes densités (figure 3.13)



(a) Image (b) Espace couleur (c) Histogramme 3D

FIG. 3.13 – Distribution des points couleurs dans l'espace RGB.

Index	Noyau	Classe représentée
1	c_1	C_1
$\vdots$	$\vdots$	$\vdots$
P	c_P	C_P
$P+1$	/	$C_{P+1} = E \setminus \cup_{i=1}^{P} C_i$

TAB. 3.2 – Palette avec classe d'exclusion (E=espace couleur, généralement $[0,1]^3$).

Il peut être intéressant de profiter de ce constat pour réduire la taille de la palette de couleur, dont dépend le temps nécessaire au calcul de la mesure de similarité. Ainsi, nous proposons de restreindre la quantification à un sous-espace de l'espace couleur. Dans certaines conditions, il se peut que l'union des $C_i = 1, \ldots, P$ ne recouvrent pas toujours l'ensemble des zones occupées de l'espace, notamment :

- lorsque la palette n'est construite que sur base d'une partie de l'image à indexer et non sur la totalité, ce qui laisse des couleurs inconsidérées et pour lesquelles aucune classe n'a été prévue.
- lorsque, lors de l'indexation d'une séquence d'images, de nouvelles couleurs apparaissent dans les images, ce qui arrive lorsque des surfaces de nouvelles couleurs entrent au sein des images durant l'acquisition de la séquence ou lorsque des variations d'éclairage viennent modifier les couleurs de la scène filmée.
- lorsqu'une couleur ne peut entrer dans une classe en raison de son trop fort éloignement par rapport au noyau de la classe.
- lorsque du bruit au capteur vient déformer l'information.

Tous les pixels devant techniquement est placé dans une classe, nous avons étendu la définition de palette pour y ajouter une classe supplémentaire dans laquelle sont placés les pixels n'appartenant pas à au recouvrement (union des $C_i = 1, \ldots, P$). Celle-ci possédera l'index P+1 et sera appelée **classe d'exclusion** C_{P+1} (Tableau 3.2).

Indexation

La deuxième étape consiste à indexer l'image, c'est-à-dire à remplacer le triplet de couleur caractérisant chaque pixel par l'indice de la classe à laquelle il appartient.

L'index associé à un triplet correspond à l'index de la classe dans la palette qui le contient. L'image indexée ne comporte dès lors plus trois plans de couleur comme l'image originale mais un seul plan d'indices pointant dans la table de correspondance indice/classe de couleur, la palette (ou la "map").

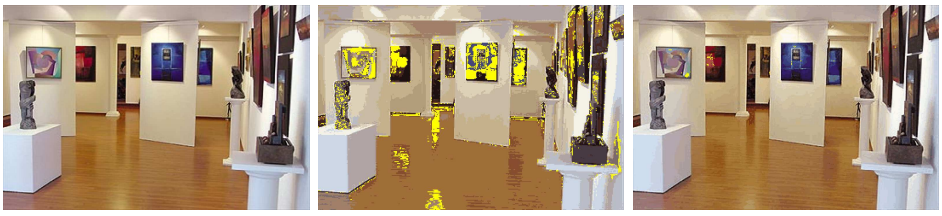
Les techniques d'indexation classiques (telles qu'implémentées dans les boîtes à outils de traitement d'image Matlab par exemple) n'incluent pas la notion de taille de classe et sélectionnent la classe dont le noyau est le plus proche, au sens d'une certaine norme. Ceci assurant que chaque pixel soit classé dans une et une seule classe.

Afin de travailler avec des quantifications restreintes, nous avons développé nos propres algorithmes travaillant avec la classe d'exclusion définie ci-dessus. Cet artifice a pour double objectif que tous les pixels soient "justement" classés une, et une seule fois. Nous avons pu démontré que les pixels injustement classés dégradaient la qualité de la mesure d'intersection.

Aussi, il est intéressant de formuler quelques remarques à propos des images indexées. La première est que la modification d'un élément de la palette implique indirectement une modification de tous les pixels de cette classe dans l'image indexée.

La seconde concerne la notion de norme ou de distance entre couleurs de pixels. Celle-ci n'est plus valable pour les pixels des images indexées : il n'y a en effet aucun sens à mesurer une distance entre des indices désignant des classes couleurs et sur lequel aucun ordre n'a été défini.

La troisième concerne le paramétrage. Il est nécessaire de trouver un bon compromis entre diminution de l'information et conservation de la qualité. Ainsi, un nombre de classes élevé permet la description de nombreuses nuances mais l'information maintenue est trop volumineuse. A l'inverse un nombre de classes trop faible ne suffit pas pour bien rendre compte de l'image (figure 3.14). Dans la section suivante, nous discutons de la mise au point d'une technique permettant de déterminer automatiquement ce paramètre en effectuant une optimisation sur un critère statistique.



(a) Image originale (35961 couleurs) (b) Image indexée (N=16, r=0.1) (c) Histogramme (N=128, r=0.1)

FIG. 3.14 – Image couleur 320 x 240 (86400 pixels) et deux indexations avec des paramétrages différents. Les zones en jaune correspondent à des groupes de pixels placés dans la classe d'exclusion.

Enfin, remarquons que la méthode exposée ici permet un contrôle plus fin de la

quantification et de l'indexation que celle qui avait été utilisée dans [1] où seul un partitionnement uniforme au paramétrage trop rigide (contraint par des questions d'arrangement en mémoire) avait été envisagé.

3.5.3 Algorithme d'indexation rapide

Réalisée naïvement, l'opération d'indexation (et plus particulièrement celle basée sur un calcul de norme) se révèle être très coûteuse. Afin de pouvoir effectuer des tests en temps réel, nous avons implémenté nos propres algorithmes d'indexation de façon très efficace : l'opération d'indexation ne consistant plus qu'en une seule indirection par pixel. Nous détaillons ici cette implémentation.

Nos prédécesseurs dans [1] utilisaient une discrétisation de l'ensemble du cube RGB en 256 classes. Ils attribuaient trois bits pour la couleur rouge, trois autres pour la couleur verte, et les deux restants pour la couleur bleu pour laquelle l'oeil humain distingue moins de nuances. Si un tel procédé constitue une découpe figée du cube RGB, elle permet d'indexer une image en effectuant de simple opérations de décalage de bits pour chacune des trois composantes couleurs. Ce calcul est nettement plus rapide que le calcul d'une norme pour chacun des pixels de l'image.

L'utilisation d'une palette de couleur spécifique nécessite une approche algorithmique différente. Nous allons recourir à la création d'un cube RGB restreint (une discrétisation du cube RGB donc) que nous conserverons en mémoire, cube qui est appelé dans la littérature anglaise "inverse colormap". A chacun des bins que ce cube contient, nous attribuerons l'index d'une des couleurs de la palette que nous avons définie pour notre image de référence. Cette "inverse colormap" constitue donc en quelque sorte une table d'indexation pour notre image.

Ce processus est trop long pour être effectué à chaque capture, car il implique un calcul de norme pour chaque élément de l'"inverse colormap". Spencer W. Thomas, qui a inspiré la toolbox matlab et plus particulièrement l'algorithme d'indexation, propose des méthodes de calculs rapides pour ces "inverse colormap", mais ces méthodes sont propres à des normes particulières (L_2 pour celle implémentée dans Matlab). Nous avons implémenté une version certes moins rapide, mais permettant de prendre en compte les vecteurs de tolérance introduits dans la section précédente.

En quelques dixièmes de seconde cette table d'indexation est générée en mémoire et elle va nous permettre de procéder d'une manière similaire à celle utilisée par nos prédécesseurs pour indexer rapidement les images dans le suivi. En effet, cette table est propre à la palette de couleurs utilisée ; tant que nous utilisons la même palette, l'"inverse colormap" reste valable.

Lors de la phase d'indexation de l'image proprement dite, les composantes RGB de l'image sont discrétisées afin de nous ramener au même nombre de bits par pixels que dans l'"inverse colormap". Les valeurs obtenues servent d'index pour accéder à un élément, un bin, de l'"inverse colormap". La valeur de cet élément constitue l'index, de la palette de couleurs, qu'il convient d'utiliser pour représenter le pixels de l'image.

Le scénario idéal serait de conserver un cube *RGB* utilisant 8 bits par canaux de couleurs, ce qui constituerait la représentation la plus fidèle de la palette de couleur que nous avions élaborée. Cependant pour effectuer l'indexation proprement dite de l'image, nous devons conserver ce cube en mémoire. En pratique, nous nous limiterons à des palettes de couleurs comptant un maximum de 256 couleurs, chacun des index de la palette ne nécessitera donc qu'un octet. Un cube contenant 8 pixels par canaux nécessite par conséquent de garder en mémoire 16 Mo. En utilisant 6 bits pour chacun des canaux de couleur, l'"inverse colormap" ne nécessitera qu'une place de 256 Ko.

Si pour des raisons d'occupations mémoires nous avons implémenté une version utilisant 6 bits par canal de couleur, ce procédé reste tout à fait valable en prenant plus de bits par canal de couleur. Il nous permet d'indexer une image de 640 pixels par 480 en moins d'une milliseconde.

3.6 Analyse et optimisation de l'histogramme couleur

Dans cette section, nous discutons de l'optimisation de la signature couleur que constitue l'histogramme couleur.

La première partie est dédiée à la capture de l'image de référence. La deuxième tranche sur la question de savoir s'il vaut mieux partitionner tout l'espace ou quantifier uniquement la partie de l'espace occupée par des points couleurs de la cible. Enfin, dans une troisième partie, nous présentons l'ébauche d'une technique permettant de déterminer automatiquement le nombre optimal de classes à utiliser pour caractériser une image en analysant celle-ci à l'aide d'une méthode statistique.

3.6.1 Niveau de détail

Les capteurs de caméra sont caractérisés par une certaine résolution qui est fixée. Dès lors, sans dispositif de zoom optique, les objets éloignés par rapport à l'objectif de la caméra sont décrits avec moins de détails que ceux qui sont plus proches. Cependant, nous voulons que l'algorithme soit capable de détecter l'objet cible aussi bien lorsqu'il est (relativement) éloigné que lorsqu'il est proche.

On peut donc se demander s'il est préférable que la capture de l'image de référence se fasse lorsque la cible est proche ou éloignée, afin de permettre le suivi de ce type de mouvement où l'éloignement n'est pas constant.

Considérons tout d'abord la première alternative. Etant prise de près, l'image de référence sera capable de rendre compte de nombreux détails de la cible. Lorsque celle-ci sera peu éloignée de la caméra, les niveaux de détails seront similaires et la détection ne rencontrera pas de problèmes majeurs. Lorsque la cible sera plus éloignée, la contribution à la mesure de similarité des pixels relatifs aux détails sera négligeable étant donné que le modèle aura également subi une réduction. Ainsi, le fait de tenir compte de petits détails que la caméra ne perçoit plus ne sera pas trop pénalisant.

Au contraire, la deuxième alternative fournit de moins bons résultats. Lorsque la cible sera éloignée de la caméra, aucun problème ne se présentera, les niveaux de détails étant du même ordre. Cependant, lorsque la cible se rapprochera de la caméra, des détails vont progressivement s'introduire dans les images de celle-ci. Si le modèle ne possède aucune information sur ces détails, leur contribution dans ce cas importante, ne peut correctement être prise en compte dans la mesure de similarité, ce qui dégrade bien évidemment sa qualité.

Cette intuition a largement été confirmée lors des nombreuses expériences de suivi que nous avons réalisées.

3.6.2 Discussion de la quantification

Deux options sont possibles lors de la construction de la quantificatio sur laquelle reposera l'histogramme.

1. Soit l'intégralité de l'espace est quantifié, l'ensemble de classe déterminé le recouvrant entièrement. L'algorithme d'indexation utilisé peut dans ce cas travailler sans classe d'exclusion.
2. Soit la quantification est limitée aux parties de l'espaces occupées par des points couleurs de l'image de référence. Toutes les couleurs ne pouvant rentrer dans une telle quantification, l'algorithme d'indexation avec classe d'exclusion *doit* être utilisé.

On peut dès lors se demander quelle options fournira le meilleur modèle de suivi. Plusieurs arguments peuvent être avancé en faveur de la seconde.

Premièrement, il est fort probable qu'un nombre élevé de couleur (autre que ceux de l'image de référence) n'apparaissent jamais dans la séquence d'image analysée. Ainsi, il est inutile de leur attribuer des classes qui resteront toujours vide, augmentant inutilement la taille de l'histogramme (et donc le temps de calcul de la mesure de similarité) pour de toute façon n'avoir aucune influence sur la mesure d'intersection.

Deuxièmement, il n'est pas nécessaire de pouvoir distinguer les couleurs ne se trouvant pas dans l'image de référence ; celles-ci peuvent toutes être considérées de la même façon et placée dans une même classe "autres couleurs". La classe "autres couleurs" de l'histogramme de la référence est toujours de valeur nulle. Dès lors, la mesure de son intersection avec l'histogramme d'une fenêtre de recherche contenant des couleurs n'étant pas dans la cible verra sa valeur diminuer de la même façon que si des classes les contenant avaient étés prévues.

Ainsi, la restriction de la quantification à la partie de l'espace occupée par des points couleur de la référence permet de diminuer la taille de l'histogramme tout en maintenant le potentiel descriptif de celui vis-à-vis de l'image de référence.

3.6.3 Optimisation

L'histogramme a initialement été choisi pour modéliser la cible par sa distribution de couleur. Celui-ci repose sur une indexation préalable par un algorithme auquel on

passe un paramètre critique : le nombre de classes. Ce dernier influence fortement les résultats du *CHS* et il conviendrait de pouvoir le déterminer automatiquement. Ainsi, la quantification de l'espace en un nombre élevé de classes permet de finement caractériser la cible mais sans offrir aucune tolérance aux éventuelles variations d'illumination de l'environnement dans lequel évolue la cible. A l'inverse, un nombre de classes trop faible ne permet pas de décrire la cible de façon suffisamment précise que pour être distinguée par rapport au reste de la scène. A ce compromis précision/tolérance, vient aussi s'ajouter une dimension de coût calculatoire. En effet, nous verrons que la mesure de similarité entre histogramme dépend directement de son nombre de classes. Il convient donc que celui - ci ne soit pas démesuré. Le caractère critique de ce paramètre s'est confirmé lors de nos nombreuses expériences.

La question qui se pose donc naturellement est de déterminer automatiquement le nombre optimal, au sens du compromis tolérance/complexité/précision, de classes qu'il faut utiliser pour construire le modèle.

Lors de nos investigations, nous avons découvert une publication de Birgé et Rozenholc, traitant de la mise au point d'une méthode pour choisir le nombre de classes à mettre dans un histogramme, sur base de l'analyse d'un échantillon des données. Leur approche, est particulièrement intéressante en ce qui nous concerne puisqu'elle ne repose, au contraire de beaucoup d'autres, sur aucune considération asymptotique, ce qui la laisse donc applicable dans le cas d'échantillons de petite taille, cas de figure qui peut se présenter lorsque la cible à modéliser est relativement petite.

Birgé et Rozenholc ont généralisé l'estimateur d'Akaike, qui est une mesure statistique permettant la sélection de modèle et basée sur le postulat suivant : si deux modèles représentent aussi bien des données, alors le modèle le plus simple sera le meilleur. Nous décrivons ici brièvement leur méthode pour ensuite exposer l'adaptation que nous en avons envisagée (la méthode étant en effet prévue pour la partition d'un espace à une dimension). Les supports et détails théoriques pouvant être consultés dans [3].

Notre objectif est donc de trouver un estimateur $\hat{f}$ de l'histogramme de la cible basé sur une partition $I_1, \ldots, I_{K^*}$ de $[0, 1]$ en K^* intervalles de même longueur. Si l'on désigne par $X_1, \ldots, X_n$ les n échantillons de la distribution f que l'on cherche à estimer, alors la valeur de K^* est donnée par

$$K^* = \text{argmax}_K(L_n(K) - \text{pénalité}(K)) \tag{3.17}$$

où $L_n(K)$ est le logarithme de la vraisemblance de l'histogramme avec K classes, donné par

$$L_n(K) = \sum_{j=1}^{K} M_j.\log\left(\frac{KM_j}{n}\right) \tag{3.18}$$

avec $M_j = \sum_{i=1}^{n} \chi_{I_j}(X_i)$
et où χ_{I_j} est la fonction caractéristique définie par

$$\chi_{I_j}(x) = \begin{cases} 1 & \text{si } x \in I_j \\ 0 & \text{sinon} \end{cases}$$

La fonction de pénalité étant définie comme

$$\text{pénalité}(K) = K - 1 + (\log(K))^{2.5},\ \forall\ K \geq 1 \tag{3.19}$$

Cette technique est conçue pour la construction d'une partition d'un espace monodimensionnel, or, il nous faut l'appliquer au partitionnement d'un (sous-espace) tridimensionnel.

Nous avons à cette fin imaginé la solution suivante. Nous appliquons la méthode à chaque dimension (R, G et B par exemple) dont résulte une partition par dimension. La combinaison de toutes les classes des trois dimensions permet de reconstruire des classes tridimensionnelles.

Afin de mettre à l'épreuve cette technique, nous avons développé des scripts Matlab réalisant cette estimation. Cependant les résultats obtenus ne sont pas vraiment satisfaisants. Sur des images artificielles, le procédé fonctionne très bien et donne un nombre de classes généralement en accord avec l'intuition d'une part, et fournit les meilleurs résultats d'expérimentation d'autre part. Par contre, sur certaines cibles complexes, le nombre de classes calculé explose.

Nous n'avons malheureusement pas eu le temps d'approfondir les recherches dans ce sens, mais nous pensons qu'il serait intéressant de généraliser le critère de façon plus subtile, peut être en travaillant au niveau de la constitution de l'échantillon de base. Cette intuition étant confortée par la découverte d'un article [20] traitant d'un problème de suivi similaire au nôtre et où cette méthode est utilisée pour déterminer le nombre de classes à utiliser pour modéliser une cible à l'aide d'un histogramme en niveaux de gris. Dans tous les tests que présentent les auteurs, les performances de suivi à l'aide du nombre de classes déterminé par la méthode de Birgé et Rozenholc sont généralement supérieurs à des suivis où le nombre de classes est fixé arbitrairement.

3.7 Mesure de similarité

Notre objectif étant de localiser une cible dans une image, il nous faut pouvoir déterminer dans quelle mesure deux images sont semblables. Cette comparaison se fera bien évidemment sur base des histogrammes couleurs introduits précédemment. A cette fin, nous introduisons dans cette section le concept de mesure de similarité qui permet de quantifier la ressemblance entre deux signatures (et donc en fin de compte, entre deux images).

La mesure de similarité la plus connue est sans doute celle de Swain et Ballard [29] proposée dans le cadre de l'indexation de base d'images et définie sur des histogrammes couleurs tridimensionnels. Appelée intersection d'histogrammes, elle est définie pour toute paire d'histogrammes H^1,H^2 :

$$d_{\cap}(H^1, H^2) = \frac{\sum_{i=1}^{N} min(H^1(i), H^2(i))}{\sum_{i=1}^{N} H^1(i)} \tag{3.20}$$

Cette fonction permet d'évaluer, sur une échelle réelle allant de zéro à un, le recouvrement d'un histogramme par l'autre. Il faut toutefois remarquer qu'elle ne constitue pas une distance au sens mathématique du terme puisqu'elle n'est pas symétrique ($\exists H^1, H^2$ tq $d_\cap(H^1, H^2) \neq d_\cap(H^2, H^1)$). Une variante introduite par Smith [30] permet de remédier à ce problème, qui n'est pas gênant en ce qui nous préoccupe puisque lorsque l'on modifie la taille de recherche, on adapte l'histogramme de référence (chapitre 5) de telle sorte que le nombre de pixels de H_1 et H_2 soit toujours identique.

Les histogrammes peuvent également être vus comme des vecteurs faisant partie d'un espace de dimension N. Diverses méthodes de calcul de distance basées sur les distances de Minkowski [30] peuvent donc être utilisées :

$$d_{L_p}(H^1, H^2) = \left(\sum_{i=1}^{N} \left| H^1(i) - H^2(i) \right|^p \right)^{\frac{1}{p}} , \ 1 \leq p < \infty \tag{3.21}$$

$$d_{L_\infty}(H^1, H^2) = max_{1 \leq i \leq N} \left| H^1(i) - H^2(i) \right| \tag{3.22}$$

Les normes L_1 et L_2 ont notamment été utilisées par Stricker pour l'indexation de contenu de base d'image [18, 28].

Il est intéressant de formuler quelques remarques à propos de ces mesures de similarité (intersection d'histogrammes et normes mathématiques)

- La mesure de distance est calculée classe par classe, ce qui implique que les deux histogrammes soient construits sur la même quantification.
- Les histogrammes conservent implicitement une information quant à la taille de l'image qu'ils représentent. Même si elle est tolérante à de faibles écarts, elle ne possède véritablement un sens que si les histogrammes représentent des images de même taille (figure 3.8).
- Une comparaison d'image sur base de ces distances est sensible aux variations d'illumination. Les images ou les signatures doivent donc subir des corrections préalables en vue d'être comparées.
- Enfin, l'avantage de ces signatures dans l'approche qui nous préoccupe est qu'elle sont peu coûteuses à calculer et que des techniques d'optimisation, dont nous discutons dans la section traitant des performances du *CHS* (3.4.1), permettent de les utiliser dans un cadre de temps réel.

Une autre approche, dans laquelle l'histogramme couleur est considéré comme la réalisation d'une variable aléatoire, consiste à ramener le problème de comparaison entre deux histogrammes à un test d'hypothèse dans lequel il faut déterminer si deux réalisations (deux histogrammes) peuvent provenir de la même distribution. Dans le cadre d'indexation de base d'image, les deux équipes ([18], page 6) ayant introduit cette vision des choses ont montré sur base d'une batterie de test (toutefois assez réduite) que les résultats fournis par leur méthode étaient meilleurs que ceux fournis par l'intersection d'histogrammes ou la norme Euclidienne. Nous n'avons malheureusement pas eu le temps d'explorer cette piste, mais proposons de ne pas l'écarter de recherches futures.

3.8 Conclusion

Dans ce chapitre, nous avons présentés l'algorithme du *CHS*, dont nous avons décrit les deux phases constitutives. La première phase consiste en l'acquisition de la cible, son indexation et la création de son histogramme couleur qui nous servira de modèle, tandis que la seconde constitue le suivi de cette cible dans une séquence vidéo. Nous avons détaillé les différentes phases du suivi qui sont l'acquisition de la scène, la formation de la surface de similarité et son analyse afin de déterminer la position de la cible. Enfin, nous avons abordé plusieurs paramètres intervenant dans l'algorithme, paramètres dont le choix influence de manière plus ou moins importante la qualité du suivi. Nous avons notamment discuté du choix de l'espace couleur, de la quantification utilisée (le nombre de classe à utiliser, l'utilisation ou non d'une classe d'exclusion, la norme utilisée, ...) et de la mesure de similarité à utiliser.

Par ailleurs, nous avons évalué le potentiel descriptif des histogrammes et mis en évidences certains problèmes épineux auxquels cet algorithme peut être confronté. Nous traiterons de manière approfondie ces différents problèmes dans les chapitres suivants. Le problème lié au changement des conditions d'éclairage sera traité dans le chapitre 4, celui lié au variation apparentes de la taille dans le chapitre 5 et celui lié à l'orientation et à la forme des motifs dans le chapitre 6.

Chapitre 4

Indépendance à l'illumination

4.1 Introduction

La perception par la caméra du spectre réfléchi par les éléments de la scène dépend de nombreux paramètres, tels que la position des objets ou de la caméra par rapport à eux, la position, l'intensité et la couleur des différentes sources lumineuses. Les variations de ceux-ci ont pour effet de modifier le résultat de l'algorithme d'indexation. Les variations peuvent être telles que certains pixels changent de classe. Il en résulte une modification de la forme de l'histogramme ensuite construit. Contrairement à ce qui était avancé dans [1], celle-ci ne se traduit généralement pas par une simple translation de l'histogramme, la relation étant beaucoup plus complexe puisque les classes de l'histogramme sont définies dans un espace couleur tridimensionnel.

Comme l'histogramme représente la distribution de couleurs d'un objet sous un illuminant particulier, l'intersection d'histogrammes perd tout son sens lorsqu'elle est appliquée à deux histogrammes représentant des images de luminosités différentes *si aucune mesure corrective n'est envisagée* (figure 4.1).

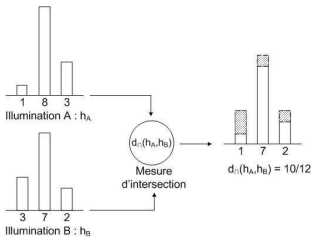


FIG. 4.1 – Intersection d'histogrammes d'une même cible vue sous deux illuminants différents.

Cependant il est attendu de l'algorithme qu'il puisse fonctionner en environnement non contrôlé et en particulier lorsque la luminosité peut varier. Il est donc nécessaire d'obtenir d'une part une certaine tolérance aux faibles fluctuations de luminosité, et d'autre part, dans la mesure du possible, une certaine indépendance à des plus fortes variations des conditions d'illumination. Différentes pistes de solutions sont envisageables.

Afin de mesurer l'efficacité des solutions que nous allons étudier, nous avons développé un critère (4.2) permettant de mesurer l'indépendance d'une signature de type histogramme (ou d'une image prétraitée) aux variations de luminosité.

Dans [1], une première solution proposée consiste à utiliser un espace séparant la chrominance de la luminance, en quantifiant celui-ci plus finement sur sa première composante et plus largement sur la seconde. Dans la deuxième section, nous explorons plus en détails la réponse que peut apporter la quantification et le choix de l'espace et montrons que, même si certains espaces donnent de meilleurs résultats que d'autres, leur seul choix ne constitue pas une solution au problème.

Des solutions "cross-bin" y ont également été explorées mais malgré ce qui était attendu, celles-ci ne fournissent que des résultats décevants, en plus d'être beaucoup plus coûteuses en temps de calcul.

Des solutions à références dynamiques ont aussi été proposées, mais il est ressorti des expérimentations qu'elles avaient tendance à dériver sur une cible totalement différente de celle de départ. L'introduction d'un mécanisme d'historique permet d'améliorer sensiblement la solution, mais celle-ci, reposant sur des indicateurs caducs, n'apporte malheureusement pas une réelle amélioration. Nous n'explorons pas plus ces deux dernières pistes.

Dans la suite du chapitre, nous aborderons différentes approches dans lesquelles un prétraitement des images est effectué afin des les rendre autant que possible indépendantes aux variations de luminosité.

L'idée d'annuler les effets de la luminosité émane directement du concept de constance de couleur qui vient lui-même de l'observation que la perception humaine de la couleur est indépendante du produit de l'illumination par la réflectance. Ainsi, des algorithmes de constance de la couleur [30] sont en cours de développement, mais malgré leur complexité croissante, ils ne fournissent encore que de modestes résultats. Nous ne nous y attardons pas.

Les méthodes de prétraitement exposées dans la suite dépendent de modèles de variation de l'illumination. Dans la troisième section, nous en présentons quelques uns, et, sur base d'une étude statistique menée dans [18], nous retenons les modèles linéaire de Brill [6] et diagonal de Fynlaison [12].

Dans les sections quatre et cinq nous exploitons de deux façons différentes ces modèles en présentant les techniques d'histogrammes ratios et de normalisation d'image,

qui permettent toutes deux de rendre l'intersection d'histogrammes indépendante de l'illumination, la seconde étant une amélioration de la première.

4.2 Critère d'indépendance

Dans la suite de ce chapitre, différentes techniques permettant d'amener une indépendance à l'illumination sont étudiées. Dans cette section, nous introduisons deux variantes d'une même mesure permettant d'une part d'évaluer l'efficacité des différentes solutions et, d'autre part, de les comparer entre elles, notamment sur plusieurs jeux de tests annexés en fin de mémoire.

Afin de ne rendre compte que des variations d'illumination, cette mesure ne sera définie que sur des images brutes, prétraitées et/ou indexées, où **seule** varie la luminosité. Cette mesure n'est donc que d'utilité expérimentale. En raison de sa définition impliquant un calcul pixel par pixel, elle ne peut pas être utilisée pour comparer deux zones d'images différentes.

La première variante est définie sur deux images indexées I et I' et mesure la proportion de pixels dont les classes dans I et I' sont différentes.

$$e_c(I^1, I^2) = \frac{\#\left\{i : c(I_i^1) \neq c(I_i^2) \text{ où } i = 1, \ldots, N\right\}}{m.n} \tag{4.1}$$

Si q_E représente une quantification de l'espace E et z un changement de luminosité, nous pouvons utiliser cette mesure e pour évaluer la tolérance apportée au changement de luminosité z par un choix d'espace E et une quantification q_E associée.

Une valeur nulle de $e_c(q_E(I), q_E(z(I))$ signifie que la quantification q_E parvient à maintenir une classification des pixels inchangée malgré le changement de luminosité z. Des valeurs proches de zéro indiquent que la quantification q_E est tolérante au changement de luminosité z. Des valeurs plus éloignées de zéro indiquent que les pixels changent de classe sous l'influence de z et que la quantification ne permet pas de le tolérer.

Remarquons que la définition d'une mesure similaire sur les histogrammes ne donnerait pas le même résultat puisqu'elle ne permettrait pas de rendre compte des éventuels échanges entre classes. Nous utilisons cette métrique dans la section suivante traitant du choix de l'espace et de sa quantification.

La seconde variante, définie sur des images brutes et prétraitées, permet de mesurer l'efficacité des résultats des algorithmes de prétraitement d'indépendance à l'illumination, qui seront présentés par la suite. Tout comme la première, cette mesure ne peut être utilisée que pour rendre compte de l'efficacité en terme d'indépendance à la luminosité que si elle est appliquée sur des images ne différant que par leur luminosité. Elle est définie pour toute paire d'images (I^1, I^2) représentant la même

scène :

$$e(I^1, I^2) = \frac{\sum_{i,j} \left| I^1_{ij} - I^2_{ij} \right|}{3.m.n} \tag{4.2}$$

De même que précédemment, si p représente un prétraitement et z un changement de luminosité, nous pouvons utiliser cette mesure e pour évaluer l'indépendance à l'illumination apportée par l'algorithme de prétraitement qui calcule p

$$e(p(I), p(z(I)) \tag{4.3}$$

Une valeur nulle de signifie que le prétraitement p parvient à annuler complètement l'effet des variations de luminosité z. Cependant, ce cas est théorique et ne se présente pratiquement jamais en situation réelle. Des valeurs proches de zéro indiquent que le prétraitement p apporte une bonne tolérance au changement de luminosité z. Des valeurs plus éloignées de zéro indiquent que les pixels changent de classe sous l'influence de z et que la quantification ne permet pas de le tolérer.

Idéalement, il nous faudrait déterminer un algorithme p tq

$$e(p(I), p(z(I)) \approx 0, \forall I, \forall z \tag{4.4}$$

Nous y revenons lors des sections quatre et cinq traitant respectivement des histogrammes ratios et de la normalisation d'images.

4.3 Espace couleur et quantification

4.3.1 Introduction

Nous avons vu qu'en cas de trop fort changement de luminosité, les pixels de l'image changent de classe, ce qui a pour effet de modifier l'histogramme et donc de perturber la mesure d'intersection avec l'histogramme de référence, restant inchangé.

Etant donné que l'histogramme dépend directement de l'espace utilisé et de la quantification associée, on peut se demander s'il existe des meilleurs (au sens de notre critère) espaces et/ou des quantifications optimales fournissant une meilleure indépendance à la luminosité.

Ce point ayant déjà été abordé en partie dans [1], nous n'en rediscuterons pas tous les aspects, mais préciserons plutôt l'impact des ces deux paramètres en considérant les cas de *RGB* et de *HSV*, qui, pour rappel, ont été sélectionnés comme espaces sur lesquels le *CHS* donne les meilleurs résultats dans des conditions diverses.

4.3.2 Discussion

Quel que soit l'espace couleur choisi pour supporter les traitements, la quantification, même si elle n'a pas la même signification d'un espace à l'autre, est un paramètre critique dont la détermination fait l'objet d'un compromis entre tolérance aux éventuelles variations et précision de description de l'image. Ainsi, une tolérance ne peut être augmentée indéfiniment sans que le modèle ne soit plus capable de caractériser l'image de la cible de façon suffisante pour l'identifier.

RGB

L'espace RGB étant fortement corrélé et les changements de luminosité a priori imprévisibles, il n'y a aucune raison de construire des classes plus ou moins étendues le long de l'une ou l'autre de ses dimensions R,G ou B. Celles-ci seront donc généralement symétriques ($p = (p, p, p)$) et admettent de faibles changements de luminosité, de même grandeur et sans direction privilégiée.

HSV

L'espace HSV a pour avantage de séparer l'information de luminance de celle de chrominance. On pourrait donc s'attendre à ce qu'une quantification appropriée, par exemple plus large dans la direction de V et plus stricte dans celles de H et de S, apporte une bonne tolérance aux changements de luminosité. Cependant, les variations dans la direction de V ne sont qu'un type de variations possibles : celles qui ne modifient ni la teinte ni la saturation. Or, il est très rare que les variations réelles ne modifient que le paramètre V seul. Ce type d'espace n'apporte donc une solution que pour un cas très particulier.

4.3.3 Conclusion

Nous venons de voir que le choix d'un espace et d'une quantification n'apporte une réponse qu'à un seul type de variation particulière. De plus, la tolérance, dépendant de la taille des classes ne peut être indéfiniment étendue puisque celles-ci doivent permettre de caractériser la cible de façon suffisamment précise. Leur taille étant restreinte, les pixels finiront toujours par changer de classe si les variations sont trop importantes. Il serait bien sûr possible de construire des classifications plus complexes, mais celles-ci n'apporteraient encore que des solutions partielles à des variations particulières et très limitées.

Bien que le choix de l'espace et de la quantification associée ne soit définitivement pas une solution au problème de dépendance à l'illumination, celui-ci ne doit tout de même pas être négligé.

En ce qui concerne l'espace, nous avons vu que les réponses apportées étaient finalement équivalentes (il suffit d'adapter la forme des classes). Comme aucun ne donne un net avantage en terme d'indépendance, nous travaillerons donc par la suite dans l'espace natif du périphérique d'acquisition (RGB dans notre cas) puisque aucune conversion n'est requise avant traitement.

Le choix de la quantification est également important puisqu'il permet de rendre l'intersection d'histogrammes robuste aux légères variations, stabilisant ainsi la détection.

4.4 Modèles de variation et transformations

4.4.1 Introduction

Une autre piste envisageable consiste à rechercher une fonction que l'on pourrait appliquer aux images dans le but d'annuler la dépendance à l'illumination. Nous partirons à cette fin des (nombreux) travaux de Finlayson dans lesquels il traite de la mise au point de transformation de normalisation.

Ceux-ci reposent principalement sur un modèle de variation de l'illumination appelé modèle de von Kries ou modèle diagonal. Bien que la justification de ces modèles soit déjà discutée dans [13, 17], nous avons trouvé intéressant de nous baser sur un travail [18] plus récent, proposant plusieurs types de modèle et détaillant une analyse statistique sélectionnant le plus réaliste. Nous résumons leur démarche et leurs conclusions dans le premier paragraphe de cette section. De plus amples détails ainsi que des résultats chiffrés d'expérimentation peuvent être consultés dans le rapport original.

Après avoir justifié le modèle de von Kries, nous étudions la transformation en ratios de couleurs et de normalisation, la première permettant d'annuler les variations spectrales et la deuxième les variations spectrales et spatiales de luminosité. Nous évaluerons également l'efficacité de cette dernière en appliquant les deux variantes de la mesure d'indépendance que nous avons développées précédemment. Enfin, nous étudions l'apport de l'utilisation de cette transformation pour le prétraitement des images en vue de leur utilisation dans la localisation *CHS*.

4.4.2 Modèles

Avant de tenter la mise au point de transformations permettant d'annuler l'effet des variations de la luminosité, il est nécessaire d'en posséder une bonne modélisation, ce dont nous traitons dans ce paragraphe.

On peut distinguer deux types de changements. Le premier est une variation de l'intensité et/ou de la couleur émise par la source. Le second est dû au déplacement de la ou des source(s) lumineuse(s), des objets et de la caméra. Comme dans [18], nous les qualifierons respectivement de changements internes et externes de la source de lumière.

Modèle de variations spectrales (sélection statistique)

Dans [18], P. Gros et al. ont testé un certain nombre de modèles de variations spectrales. Chacun d'eux décrit la transformation d'un pixel $p = (r, g, b)$ en un autre pixel $p' = (r', g', b')$. Si l'on choisit de noter

- la translation simple par $t = (t, t, t)$ où $t \in \Re$,
- la translation générale par $T = (t_1, t_2, t_3)$ où $t_{i=1,2,3} \in \Re$,
- la matrice diagonale $D = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & \gamma \end{pmatrix}$ où $\alpha, \beta, \gamma \in \Re$

– la matrice 3×3 quelconque M
les modèles transformant p en p' peuvent s'écrire comme

Numéro du modèle	Equation	Nom
1	$p' = p$	Identité
2	$p' = p + t$	Translation simple
3	$p' = p + T$	Translation
4	$p' = \alpha p$	Scalaire
5	$p' = \alpha p + t$	Scalaire avec translation simple
6	$p' = \alpha p + T$	Scalaire avec translation
7	$p' = Dp$	Diagonal
8	$p' = Dp + t$	Diagonal avec translation simple
9	$p' = Dp + T$	Diagonal avec translation
10	$p' = Mp$	Linéaire
11	$p' = Mp + t$	Linéaire avec translation simple
12	$p' = Mp + T$	Linéaire avec translation

L'objectif principal de leur étude est de classer les modèles suivant leurs résultats sur des images réelles. Il est certain que les modèles possédant le plus de paramètres sont ceux qui fournissent les meilleures approximations. Cependant, l'utilisation d'une méthode basée sur une mesure de l'erreur et un test statistique permet de décider si un modèle diagonal avec translation est plus ou moins réaliste qu'un modèle linéaire, ce qui n'est a priori pas évident. Pour chacun des modèles retenus, ils ont également cherché à déterminer l'ensemble des paramètres nécessaires. A cette fin, ils ont déterminé si chaque paramètre avait une valeur statistiquement différente de zéro ou si, au contraire, zéro était une bonne approximation de la valeur théorique. Ceci permet de juger si le paramètre capture une information relative au bruit ou au signal (en considérant que les erreurs sont de moyenne nulle)

Méthode d'évaluation des modèles. L'évaluation et la comparaison des modèles ont été réalisées sur un ensemble d'images d'une même scène, capturées avec une caméra et une source lumineuse fixe, où seule l'intensité de celle-ci variait.

Les paramètres des modèles sont ensuite calculés de plusieurs façons. Deux algorithmes d'estimation ont été utilisés : la méthode de décomposition en valeurs singulières, fournissant une approximation linéaire aux moindres carrés des modèles et sa version robuste, dite des moindres carrés médians. Les algorithmes sont chaque fois appliqués en tenant ou ne tenant pas compte des pixels saturés. La moyenne, le maximum et le médian des erreurs entre les pixels d'une image et ceux de l'autre corrigée à l'aide des différents modèles estimés sont également calculés.

Test des paramètres estimés. Afin de vérifier que les paramètres estimés capturent bien une information et pas du bruit, une méthode (proposée par Florou [14]) basée sur un test statistique a été utilisée. Les erreurs y sont supposées de moyennes nulles. Pour chaque paramètre p_i, un intervalle de confiance à $\alpha\%$ (ie : un intervalle pour lequel on est sûr à $\alpha\%$ qu'il contient la valeur réelle du paramètre) autour de la

valeur estimée de p_i est calculé. Le rayon de cet intervalle, pour un taux de confiance α est donné par

$$\sigma^2(p_i) = \sqrt{\chi^2(\alpha, m)}\sqrt{\frac{f}{n}}\sqrt{C_{ii}} \tag{4.5}$$

où
- $\chi^2(\alpha, m)$ est le quantile de la distribution χ^2 pour un taux de confiance de $\alpha\%$ et m degrés de liberté, ce nombre étant dans notre cas le nombre de paramètres estimés ;
- f est la somme des erreurs sur tous les pixels, et n est le nombre de pixels dans chaque image ;
- C_{ii} est la variance du paramètre p_i

Ainsi, si un paramètre p_i est compris entre $-\sigma^2(p_i)$ et $\sigma^2(p_i)$, on peut alors estimer que 0, qui appartient à la région de confiance aurait fourni une estimation aussi valable, et le paramètre p_i est jugé non significatif. Dans le cas contraire, il est jugé significatif.

Expériences et résultats. Dans une première expérience, les méthodes de calcul sont comparées sur un jeu de six images (disponible dans [18], page). Il en ressort que la méthode des moindres carrés médians avec suppression des pixels saturés est celle qui donne les résultats de meilleure qualité.

Cette dernière méthode est utilisée dans une seconde expérience où les modèles sont mis en confrontation sur le jeu complet des six images. Pour chaque paire reprenant la première image et une des cinq autres de luminosités différentes, l'erreur moyenne, maximale et médiane entre ces dernières et les corrections effectuées via les modèles estimés sont calculées. Cette série permet d'une part d'évaluer les modèles pour des dispersions de pixels différentes dans l'espace couleur. D'autre part, on peut évaluer dans quelle mesure se dégradent les résultats lorsque croît la différence d'illumination, ce qui permet d'une part d'éliminer certains modèles et d'autre part de classer entre eux ceux qui sont retenus.

Les conclusions de cette expérience sont les suivantes, les résultats chiffrés pouvant être consultés à la page 20 de [18].
- Les modèles ne faisant intervenir qu'une translation fournissent de mauvais résultats qui en outre se dégradent très fortement lorsque les variations sont plus importantes. Les modèles 1, 2 et 3 sont donc définitivement écartés.
- Pour chacun des groupes de modèles scalaires (4,5,6) , diagonaux (7,8,9) et linéaires (10,11,12), l'importance des termes de translation croît avec la différence de luminosité.
- Pour chacun des groupes ayant même terme de translation, c'est-à-dire (4,7,10) ,(5,8,11) et (6,9,12), on peut étudier l'influence du nombre de paramètres dans la partie multiplicative, celle-ci étant d'autant plus marquée que la différence de luminosité est grande. Les modèles à un seul paramètre voient leurs performances se dégrader et leur écart par rapport aux autres modèles augmenter lorsque la différence entre images croît.

- Même s'il existe une différence entre les modèles diagonaux et linéaires, celle-ci est très faible et reste constante par rapport à la différence de luminosité.
- Le modèle diagonal avec translation (9) donne globalement des résultats supérieurs ou équivalents aux modèles 10 et 11 et très légèrement inférieurs au modèle 12, beaucoup plus complexe.

Le modèle diagonal avec translation (9) semble donc être le meilleur modèle au sens du compromis qualité/complexité. Par rapport à celui-ci, le modèle diagonal de Finlayson reste valide pour autant que les variations ne soient pas trop importantes.

Une dernière expérience vise à déterminer la dégénérescence des modèles par rapport à la taille d'image considérée. La signification des paramètres du modèle 12 a ainsi été testée en considérant des sous-images de taille croissante. Deux éléments intéressants en ressortent :

- Les paramètres diagonaux sont bien plus significatifs que les non diagonaux (ce qui est en faveur du modèle de von Kries utilisé par Finlayson)
- Même pour de petites sous-images, les paramètres de translation apparaissent déjà significatifs, ce qui ajoute une réserve par rapport au modèle de Finlayson.

Modèle de variations spatiales

Le modèle de variation spatiales peut être dérivé du modèle généralisé de formation de l'image 2.5. On voit dans ces équations que la dépendance de la lumière réfléchie à la configuration spatiale (positions relatives de la source, de la surface et de l'observateur) est modélisée par une multiplication par un même scalaire sur les trois canaux. La lumière diffuse peut également être approximée avec un même facteur proportionnel sur les trois composantes. Ainsi, les réponses d'un même capteur dans deux configurations géométriques c_1 et c_2 de la surface et de la source sont liées par les relations scalaires suivantes :

$$R_2 = \delta \; R_1 \tag{4.6}$$
$$G_2 = \delta \; G_1 \tag{4.7}$$
$$B_2 = \delta \; B_1 \tag{4.8}$$

La quantité δ varie bien sûr en fonction de la position sur la surface, et est donc différente pour chaque pixel de l'image.

Conclusion

Dans ces paragraphes, nous avons discuté du choix de modèles caractérisant les variations spectrales et externes de la source. Les expériences de P. Gros et al. concluent sur le choix du modèle diagonal avec translation en ce qui concerne les variations spectrales. Nous retiendrons cependant le modèle diagonal de von Kries, plus simple et qui donne des résultats sensiblement identiques lorsque les images ne sont pas trop grandes et que les différences de luminosité ne sont pas trop importantes, ce qui rentre dans le cadre de l'application que nous envisageons. Le modèle de variation externe est, quant à lui, directement dérivé du modèle général de formation de l'image. Dans

le paragraphe suivant, nous présentons des technique de normalisation (permettant théoriquement d'atteindre une certaine indépendance aux variations de luminosité), toutes basées sur les modèles que nous venons de discuter. Nous envisagerons ensuite une modification de l'algorithme du *CHS* de telle sorte que celui-ci soit, si pas complètement indépendant aux variations de luminosité, au moins beaucoup plus tolérant qu'il ne l'est dans sa version originale.

4.4.3 Transformation pour invariance

Dans cette section, nous discutons de la mise au point de transformations invariantes aux variations de luminosité, celles-ci étant basées sur les modèles exposés plus haut.

Histogrammes ratios

Les histogrammes ratios [15, 12] sont une une technique adressant le problème d'invariance aux variations spectrales de la source, en se basant sur le modèle diagonal des variations d'illumination. En remplaçant la valeur des pixels par le ratio des valeurs des pixels voisins[1], cette méthode permet d'éliminer les paramètres du modèle diagonal, c'est-à-dire la dépendance aux changements internes de luminosité.

Considérons deux pixels voisins p_1 et p_2 vus sous un même illuminant a et notons la réponse des capteurs associés (R_1^a, G_1^a, B_1^a) et (R_2^a, G_2^a, B_2^a). En considérant le modèle diagonal comme valide, un changement d'illumination, établissant une illumination b, transforme ces réponses en $(R_1^b, G_1^b, B_1^b) = (\alpha R_1^a, \beta G_1^a, \gamma B_1^a)$ et $(R_2^b, G_2^b, B_2^b) = (\alpha R_2^a, \beta G_2^a, \gamma B_2^a)$.

Calculons maintenant les ratios des réponses des capteurs pour ces pixels voisins, vus sous deux illuminations différentes. Sous la lumière a, on a

$$\frac{R_1^a}{R_2^a}, \quad \frac{G_1^a}{G_2^a}, \quad \frac{B_1^a}{B_2^a} \tag{4.9}$$

et sous la lumière b les ratios deviennent

$$\frac{R_1^b}{R_2^b} = \frac{\alpha R_1^a}{\alpha R_2^a}, \quad \frac{G_1^b}{G_2^b} = \frac{\beta G_1^a}{\beta G_2^a}, \quad \frac{B_1^b}{B_2^b} = \frac{\gamma B_1^a}{\gamma B_2^a} \tag{4.10}$$

On voit dans les équations 4.10 que les facteurs α, β et γ se simplifient lors du passage aux ratios, de telle sorte que ceux-ci soient identiques dans les deux conditions d'illuminations. Les ratios des pixels voisins sont donc invariants aux changements de luminosité (dans les limites du modèle diagonal).

De par leur définition, les ratios des pixels voisins internes à des surfaces uniformes (où tous les pixels ont la même valeur) sont tous de valeur unitaire, n'apportant aucune information utile à propos de l'image. Aux frontières délimitant ces surfaces, les valeurs des pixels voisins sont différentes et les ratios prennent alors des valeurs différentes de un. Comme les ratios ne maintiennent qu'une information utile aux frontières des différentes surfaces, ceux-ci peuvent grossièrement être assimilés à une

[1] Le voisinage est définit à l'aide d'un élément structurant,

forme de détection de contours.

Dans l'optique de modélisation de cible qui nous concerne, les images ratios pourraient être utilisées pour former des histogrammes eux-mêmes indépendants aux variations de luminosité. Il a été montré dans [15] que des histogrammes ainsi construits offraient de bonnes performances de reconnaissance et de récupération d'images dans des bases de données. Ceux-ci devraient donc entrer dans le cadre hypothétique que nous nous sommes fixé. Cependant, cette approche présente certaines limites. La première est la grande instabilité des ratios sur des images bruitées, particulièrement pour les faibles valeurs de pixel. Des légères variations dans la valeur des pixels peuvent engendrer des changements relativement importants dans les valeurs ratios. La seconde est que cette technique ne fournit qu'une indépendance aux variations spectrales.

Coordonnées de chromaticité

La transformation en coordonnées de chromaticité permet d'apporter une réponse au problème de variation externe de la source et se base sur le modèle de variation externe (2.5)

Au vu des équations, il est très simple de supprimer la dépendance de la réponse des capteurs aux variations spatiales en éliminant le facteur multiplicatif δ en divisant chaque canal R, G et B de chaque pixel par la somme des réponses de ces trois mêmes canaux :

$$r = \frac{R}{R+G+B} \tag{4.11}$$

$$g = \frac{G}{R+G+B} \tag{4.12}$$

$$b = \frac{B}{R+G+B} \tag{4.13}$$

Le passage aux coordonnées (r, g, b), couramment appelées coordonnées de chromaticité, est le moyen le plus simple d'obtenir des quantités invariantes aux changements externes.

La figure 4.2 montre l'effet de l'application de cette transformation. Dans l'image de gauche, il est clair que la lumière réfléchie en un point dépend de la position de ce point par rapport à la source de lumière. L'application de la transformée en coordonnée de chromaticité, dont résulte l'image de droite, supprime cette dépendance, ce qui a pour effet de supprimer les ombres.

Normalisation d'image (Comprehensive Image Normalisation)

Dans cette section, nous présentons la "Comprehensive Image Normalisation" développée par Finlayson. Cette technique de normalisation repose toujours sur les

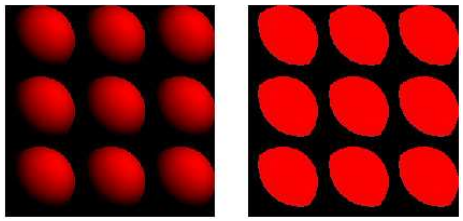


FIG. 4.2 – Effet de l'application de la transformée en coordonnées de chromaticité (droite) à une image avec des ombres (gauche).

modèles diagonal de variation interne et scalaire de variation externe de l'illumination introduit plus haut et a l'avantage de combiner les deux pour construire des images indépendantes aux deux types de changements, tout en se débarrassant des limitations inhérentes à la méthode ratios. Afin de faciliter les notations, nous représentons une image vue sous une illumination a par une matrice $N \times 3$, notée I^a. Chaque ligne de la matrice correspond à un pixel de l'image, tandis que les trois colonnes correspondent aux canaux R, G et B de tous les pixels.

$$I^a = \begin{bmatrix} R_1^a & G_1^a & B_1^a \\ \vdots & \vdots & \vdots \\ R_i^a & G_i^a & B_i^a \\ \vdots & \vdots & \vdots \\ R_N^a & G_N^a & B_N^a \end{bmatrix} \tag{4.14}$$

L'image d'une même scène dans laquelle des variations spectrales et externes établissent les conditions d'illumination b s'écrit alors comme

$$I^b = \begin{bmatrix} R_1^b & G_1^b & B_1^b \\ \vdots & \vdots & \vdots \\ R_i^b & G_i^b & B_i^b \\ \vdots & \vdots & \vdots \\ R_N^b & G_N^b & B_N^b \end{bmatrix} = \begin{bmatrix} \alpha R_1^a \delta_1 & \beta G_1^a \delta_1 & \gamma B_1^a \delta_1 \\ \vdots & \vdots & \vdots \\ \alpha R_i^a \delta_i & \beta G_i^a \delta_i & \gamma B_i^a \delta_i \\ \vdots & \vdots & \vdots \\ \alpha R_N^a \delta_N & \beta G_N^a \delta_N & \gamma B_N^a \delta_N \end{bmatrix} \tag{4.15}$$

Les facteurs d'échelle α, β et γ modélisent les changements internes et les facteurs $\delta_1, \ldots, \delta_N$ modélisent les variations spatiales.

Nous avons vu dans le paragraphe sur la transformée en coordonnées de chromaticité que la division de la réponse d'un canal par la somme des réponses des trois canaux d'un pixel permet de le rendre indépendant des variations spatiales. De la même façon, une invariance aux changements internes peut être apportée pour chaque pixel en divisant la réponse de chacun de ses canaux par la moyenne de la réponse des canaux correspondant de tous les pixels :

$$\frac{\alpha R_i^a}{\frac{1}{N}\sum_{i=1}^{N} \alpha R_i^a} = \frac{R_i^a}{\frac{1}{N}\sum_{i=1}^{N} R_i^a} \tag{4.16}$$

$$\frac{\alpha G_i^a}{\frac{1}{N}\sum_{i=1}^{N} \alpha G_i^a} = \frac{G_i^a}{\frac{1}{N}\sum_{i=1}^{N} G_i^a} \tag{4.17}$$

$$\frac{\alpha B_i^a}{\frac{1}{N}\sum_{i=1}^{N} \alpha B_i^a} = \frac{B_i^a}{\frac{1}{N}\sum_{i=1}^{N} B_i^a} \tag{4.18}$$

Donc, si nous voulons qu'une image I soit indépendante des variations spatiales, il nous suffit de diviser les éléments de chaque ligne par la somme des éléments dans la ligne. Si nous voulons que I soit indépendante des variations spectrales, il nous suffit de diviser les éléments de chaque colonne par la moyenne de tous les éléments cette colonne.

Le problème est maintenant de construire une transformation qui établit ces deux propriétés de façon simultanée. Un rapide examen des équations indique que l'application successive des deux transformations n'établit pas les deux invariants. Il a cependant été prouvé dans [16] que l'application itérative de ces deux transformations convergeait (en quelques itérations seulement) vers un point fixe et que ce point fixe présentait bien la conjonction des deux propriétés d'indépendance attendues. L

L'algorithme itératif suivant permet d'établir les propriétés en question.

Soit $(r_i^{(p)}, g_i^{(p)}, b_i^{(p)})$ un pixel de l'image à l'étape p. La valeur du pixel à l'étape $p+1$ est

$$r_i^{(p+1)} = \frac{N r_i^{(p)}}{r_i^{(p)} + g_i^{(p)} + b_i^{(p)}} \left(\sum_{j=1}^{N} \frac{r_j^{(p)}}{r_j^{(p)} + g_j^{(p)} + b_j^{(p)}} \right)^{-1} \tag{4.19}$$

$$g_i^{(p+1)} = \frac{N g_i^{(p)}}{r_i^{(p)} + g_i^{(p)} + b_i^{(p)}} \left(\sum_{j=1}^{N} \frac{g_j^{(p)}}{r_j^{(p)} + g_j^{(p)} + b_j^{(p)}} \right)^{-1} \tag{4.20}$$

$$b_i^{(p+1)} = \frac{N b_i^{(p)}}{r_i^{(p)} + g_i^{(p)} + b_i^{(p)}} \left(\sum_{j=1}^{N} \frac{b_j^{(p)}}{r_j^{(p)} + g_j^{(p)} + b_j^{(p)}} \right)^{-1} \tag{4.21}$$

Cette procédure de normalisation établit des résultats aux propriétés plutôt remarquables.

Premièrement, et au contraire de la transformée en image de ratios, le résultat constitue encore une image, comportant une information utile sur les zones uniformes(figure 4.3).

La grande similarité des images de la figure 4.3 démontre bien leur caractère indépendant, ce dont nous nous sommes assurés à l'aide du critère d'indépendance développé ci-dessus (4.2).

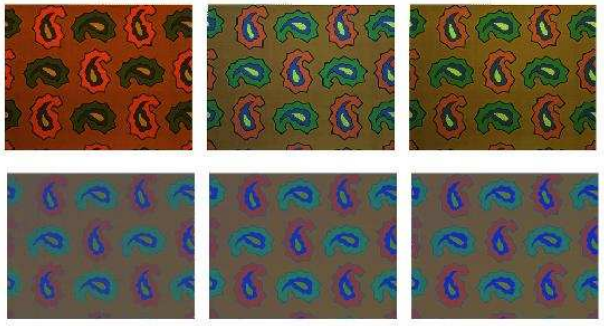


FIG. 4.3 – Effet de l'application de la transformation "Comprehensive Image Normalisation" (bas) à des images d'une même scène prise dans des conditions d'illuminations différentes. (haut) (source [12])

Dans l'objectif de l'intégrer à l'algorithme du *CHS*, nous avons expérimentalement vérifié que le pic de la surface de similarité se distinguait aussi bien en utilisant des images normalisées que des images brutes. Ceci signifie que les histogrammes construits sur bases d'images normalisées sont aussi discriminants que ceux construit sur des images brutes.

L'algorithme itératif converge généralement en quelques pas seulement.

- Sur des images en 640 par 480, l'algorithme (implémenté tel quel en Matlab) converge généralement en un nombre de pas compris entre 5 (en 2,93 secondes) et 9 (en 5,21 secondes)pas, dépendant de la luminosité globale. Plus l'image est sombre, plus l'algorithme à besoin d'itération pour converger. Ce test correspond au cas de la normalisation d'une scène complète.
- Sur des images en 100 par 100, l'algorithme converge généralement en un nombre de pas compris entre 3 (en 42ms) et 4 (en 54ms) pas. Ce test correspond au cas de la normalisation d'une fenêtre de recherche.

Enfin, signalons que des méthodes directes [18, 16] existent pour calculer cette transformation. Celles-ci fournissent les mêmes résultats, mais de façon plus efficace, ce qui est intéressant dans un contexte de temps réel.

4.4.4 Algorithme du *CHS* et normalisation

Intégration

Conceptuellement, l'intégration de la solution de normalisation à l'algorithme du *CHS* est triviale. Un prétraitement (figure 4.4, étape (1)) dans lequel sont normalisées dans l'espace *RGB* l'image de référence et la fenêtre de recherche est d'abord effectué. Cette opération s'effectue de façon indépendante pour les deux images. Ces images normalisées sont ensuite indexées (étape (2)) pour construire leurs histogrammes respectifs (étape (3)). Ces derniers sont finalement comparés (étape (4)) à l'aide de la mesure de similarité de Swain et Ballard.

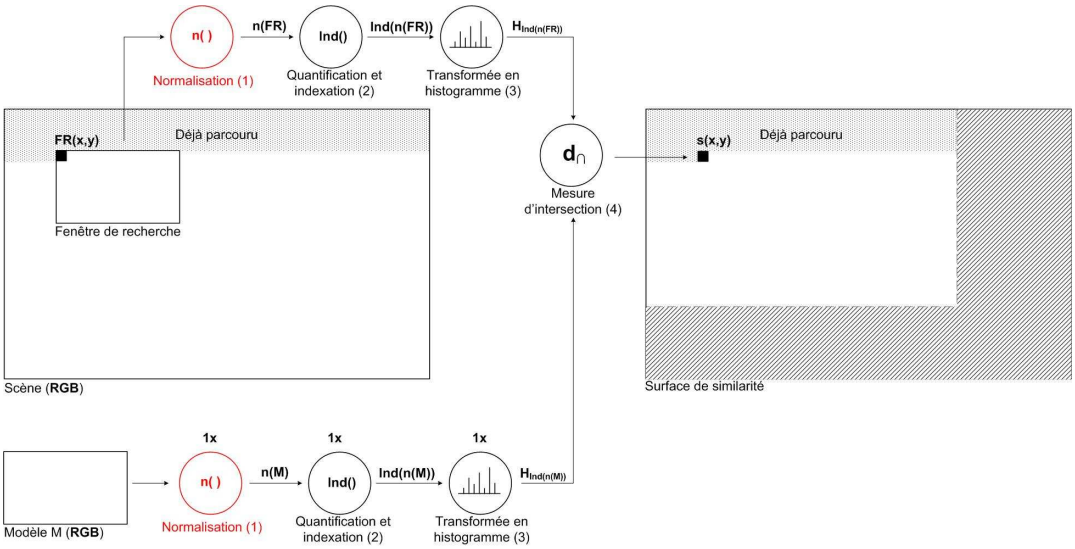


FIG. 4.4 – Calcul de la surface de similarité

Evaluation

Afin d'évaluer l'indépendance aux variations d'illumination apportée par la normalisation, nous avons calculé les valeurs de e, de e_c (section 4.2) et de similarité entre des images de luminosités différentes d'un jeu de test limité , non normalisées d'une part, et normalisées d'autre part. Il est ressorti des ces expériences que

- les valeurs des mesures e et ec calculées entre images non normalisées étaient toujours inférieures à celles calculées entre images normalisées
- la valeur de similarité calculée entre images non normalisées était toujours inférieure à celle calculée entre images normalisées

Ces deux résultats indiquant donc une amélioration. Toutefois, afin de s'assurer du véritable apport de la normalisation dans tous les cas, il conviendrait de répéter l'expérience sur un jeu de test plus conséquent que celui que nous avons utilisé.

Par ailleurs, même si conceptuellement l'intégration de la normalisation au *CHS* est évidente, il n'en va pas de même sur le plan opérationnel.

La valeur de chaque triplet de l'image normalisée dépend de l'intégralité des triplets RBG de pixels de l'image brute (équations 4.21). Dès lors, il n'est pas équivalent de normaliser la fenêtre de recherche (cas dans lequel seuls ses pixels sont pris en compte) ou de normaliser l'entièreté de la scène pour ensuite en extraire la fenêtre de recherche (cas dans lequel les pixels de la fenêtre de recherche *et du reste de la scène* sont pris en compte).

Donc, pour que l'image de référence normalisée soit comparable aux différentes fenêtres de recherche normalisées, il faut que celles-ci soient normalisées indépendamment de la scène, comme l'est la référence. Ceci empêche malheureusement de tirer parti de l'optimisation par différence d'histogrammes qui permettait de rendre le coût du calcul de la surface de similarité suffisamment faible que pour être utilisé en temps réel.

4.4.5 Conclusion

Dans cette section, nous avons abordé l'étude de transformations permettant de traiter les images *RGB* pour annuler leur dépendance aux conditions d'illumination.

Nous avons d'abord commencé par discuter des modèles de variations des conditions d'illumination sur lesquels reposent ces transformations. Cette discussion nous a amener à retenir :

- le modèle diagonal (ou de von Kries) pour décrire la dépendance des images aux variations spectrales,
- un modèle à coefficient variable, fonction de la position dans l'image, pour décrire la dépendance des images aux variations spatiales.

Nous avons ensuite présenté et analysé trois transformations proposées par Finlayson.

La première est la transformée en image de ratios, permettant d'annuler la dépendance aux variations spectrales. Nous avons vu que les images ratios délivrées par celle-ci pouvaient se révéler très instables et ainsi perturber l'intersection d'histogrammes. Nous ne l'avons donc pas retenu.

La deuxième est la transformée en coordonnées de chromaticité, permettant d'annuler la dépendance aux variations spatiales. A elle seule, cette transformation n'est pas non plus suffisante puisqu'elle ne prend pas en compte les variations spectrales.

Nous avons enfin présenté la "Comprehensive Normalisation" de Finlayson, qui permet d'annuler la dépendance aux deux types de variations de façon simultanée. L'implémentation de la version itérative de son algorithme nous a permis de procéder à plusieurs vérifications. Nous nous sommes d'abord assuré, à titre de validation de la théorie, du caractère indépendant des images normalisées. Dans une optique d'intégration au *CHS*, nous avons ensuite vérifié par analyse de la surface de similarité que les histogrammes construit sur des images normalisées étaient aussi discriminant que des histogrammes construit sur des images brutes. Enfin, des tests de performance sur l'implémentation Matlab montre qu'il serait possible, moyennant implémentation plus efficace, d'appliquer la "Comprehensive normalisation" en temps réel. Rappelons également qu'il existe des versions directes de l'algorithme, que nous n'avons pas eu le temps d'expérimenter, mais qui permettraient probablement d'établir le même résultat de manière plus efficace.

Finalement, nous avons proposé une version modifiée du *CHS* intégrant cette transformation de normalisation. Malheureusement, nous sommes arrivé à la conclusion qu'il était nécessaire de normaliser indépendamment des autres chaque fenêtre de recherche, ce qui empêche d'utiliser l'optimisation par différence d'histogramme rendant le *CHS* applicable en temps réel. Des recherches dans ce sens devraient donc encore êtres effectuées.

4.5 Conclusion

Nous avons vu que la perception des objets et de leur environnement dépendait de façon complexe de la position et du spectre des sources en présence. Des variations de celle-ci induisent une perception différente de l'image qui se répercute sur la mesure de similarité.

Tout au long de ce chapitre, nous avons tenté d'apporter une réponse à ce problème de dépendance du *CHS* aux conditions d'illumination.

Dans un premier temps, nous avons travaillé sur le choix de l'espace et sur sa quantification afin d'amener une certaine *tolérance* à de légères fluctuations de luminosité. Nous sommes arrivé à la conclusion que le choix de l'espace n'était finalement pas très important en ce qui concerne cet aspect puisqu'il n'amène des améliorations que pour certains types de variation particulières (exemple d'HSV). C'est principalement le choix de la quantification qui détermine la tolérance de l'algorithme. Toutefois, ceci n'apporte qu'une solution à des variations limitées.

Dans un second temps, nous avons voulu accroître la tolérance apportée par la quantification pour essayer de rendre le *CHS* aussi *indépendant* que possible à de plus grandes variations de luminosité. Nous avons étudié des transformations de normalisation et retenu la "Comprehensive Normalisation" permettant d'annuler la dépendance aux variations spectrales (modèle de von Kries) et spatiales (modèle multiplicatif). Différentes expériences nous ont permis d'une part de valider les résultats de ces transformations et d'autre part de montrer la possibilité et l'intérêt de leur intégration au *CHS*.

Enfin, nous avons proposé une version étendue comparant non plus des images brutes, mais des images normalisées de façon indépendante. Si les résultats sont intéressants au plan théorique, des optimisations restent à réaliser pour envisager son utilisation en temps réel.

Chapitre 5

Estimation de la taille

5.1 Introduction

L'algorithme de localisation doit être capable d'assurer le suivi de tous les mouvements relatifs de la cible et de la caméra, pour autant que certaines contraintes sur l'angle de vue soient respectées. Dans la plupart des applications, il est en particulier désirable de pouvoir continuer à suivre la cible lorsque celle-ci se rapproche ou s'éloigne de la caméra. Ce type de mouvement se traduit, dans la séquence d'images rendue par la caméra, comme une modification de taille apparente.

L'algorithme de base du *CHS* utilise, tout au long d'un suivi sur une séquence, une fenêtre de recherche et un histogramme de référence fixés. Construits à l'initialisation, ceux-ci ne sont valables que pour la taille initiale, c'est-à-dire uniquement pour la distance séparant la caméra de la cible au moment de la capture de cette dernière. Il est donc nécessaire de connaître a priori (avant chaque itération) sa taille afin de pouvoir adapter la fenêtre de recherche d'une part, et le modèle de suivi d'autre part en vue de calculer la surface de similarité. Nous étudierons cette gestion des re-dimensionnements en section 5.2.

L'algorithme du *CHS* est capable de localiser sa cible même si l'on ne dispose que d'une approximation[1] de la taille de la cible. Cependant, pour de trop grandes variations, la tolérance de l'algorithme est dépasséee et la détection perd tout son sens. La taille doit alors être recalculée. L'algorithme du *CHS* doit donc être étendu (section 5.3) pour explicitement prendre en compte ce paramètre critique que nous dénommerons facteur d'échelle[2] (noté r). Sa valeur est utile pour améliorer la précision et la fiabilité de la localisation.

En dehors des ces considérations purement inhérentes au traitement d'images, la connaissance de ce facteur d'échelle peut se révéler utile dans un cadre d'asservissement visuel pour estimer la distance séparant la cible de la caméra puisque celle-ci lui est corrélée (figure 5.1).

[1] L'intersection d'histogramme tolère en effet de légère variation de la taille de la cible.

[2] Le facteur d'échelle (noté r) est le rapport entre la taille (en pixels) de l'image cible et la taille de l'image de référence.

Diverses solutions ont déjà été étudiées telles que l'analyse d'intervalles développée dans [1] ou l'analyse d'empreinte introduite dans [9] et brièvement testée par André et Frippiat dans leur mémoire. Dans leurs conclusions sur les résultats fournis par l'empreinte, ceux-ci mettent en évidence certains problèmes mais encouragent tout de même la poursuite de son développement, ce que nous ferons dans la section 5.4. Nous verrons que celle-ci n'apporte pas une solution suffisante et proposerons une technique différente basée sur la recherche d'un minimum sur une famille de courbe en section 5.5.

5.2 Gestion des re-dimensionnements

Les mouvements modifiant l'éloignement relatif de la cible par rapport à la caméra se traduisent dans la séquence d'images rendue par la caméra par une modification de taille apparente de l'image cible, dont il faut explicitement tenir compte dans la détection.
Le MIPS propose de modéliser cette relation à l'aide de l'équation suivante

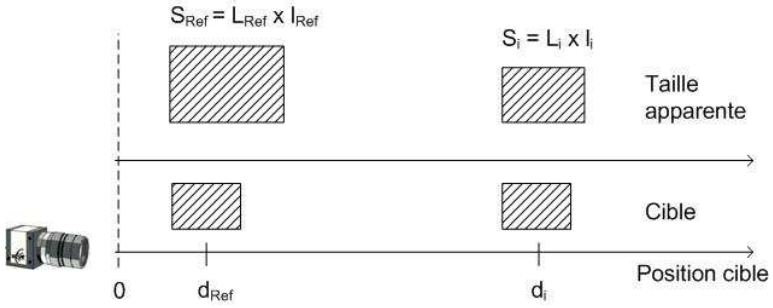


FIG. 5.1 – Taille apparente de l'objet cible (à la capture de la référence et à la i^{e} image de la séquence.

$$\frac{S_i}{S_{Ref}} = \frac{d_{Ref}^2}{d_i^2} \tag{5.1}$$

où

- S_i = surface de la cible à l'image i de la séquence
- d_i = distance cible/caméra à l'image i de la séquence
- S_{Ref} = surface de la référence, au moment de la capture de la référence.
- d_{Ref} = distance cible/caméra ,au moment de la capture de la référence.

Cette relation, basée sur des lois d'optique géométrique élémentaire, permet de définir le facteur d'échelle entre la taille de la cible à détecter et la taille de l'image de référence.

$$r_i = \sqrt{\frac{S_i}{S_{Ref}}} = \frac{d_{Ref}}{d_i}. \tag{5.2}$$

L'utilité de ce facteur est double. D'une part, il permet de remettre à l'échelle la fenêtre de recherche utilisée pour la construction de la surface de similarité. Ainsi, si

l'on suppose que la dimension initiale de la fenêtre est de $L_{Ref} \times l_{Ref}$, la dimension de la fenêtre de recherche à l'image i de la séquence sera égale à

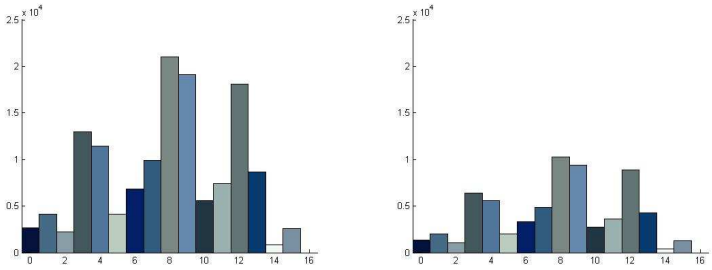
$$L_i \times l_i = (L_{Ref} \times r_i) \times (l_{Ref} \times r_i). \tag{5.3}$$

Remarquons aussi que ce type de correction pourrait immédiatement se généraliser à des contours plus complexes, de types polygonaaux par exemple, permettant de saisir des formes plus complexes.

D'autre part, le remise à l'échelle de l'histogramme de référence H_{ref} à celle de la fenêtre de recherche i se fait par simple multiplication scalaire (figure 5.2). L'histogramme H_i de cette dernière est donc calculée à l'aide de l'équation :

$$H_i = r_i^2 \; H_{ref} \tag{5.4}$$

(a) Image originale (b) Image réduite à 70%



(c) Histogramme de l'image originale (d) Histogramme de l'image réduite à 70%

FIG. 5.2 – Remise à l'échelle par multiplication scalaire (histogramme de 16 classes). Les deux histogrammes sont de tailles différentes mais dénotent des proportions de couleurs identiques.

L'établissement de cette relation suppose implicitement que les proportions de couleurs soient conservées par la transformation de remise à l'échelle. En réalité, ce n'est pas exactement le cas puisque la caméra, possédant un capteur de résolution fixée, rend compte d'un niveau de détail variant avec la distance qui la sépare de l'objet cible. Cependant, les écarts observés par rapport à ce modèle multiplicatif sont minimes et nous considérerons donc comme valide.

Remarquons que cette relation permet une remise à l'échelle très efficace du modèle, en temps proportionnel à la taille de l'histogramme (qui est constante), ce qui

est heureux dans la perspective d'une application temps réel.

5.3 Extension générale du *CHS* pour la gestion des redimensionements

Dans cette section, nous présentons une extension possible du *CHS* (que nous appelons par la suite *rCHS*) permettant de gérer explicitement les variations de taille. Dans un premier temps, nous nous sommes volontairement abstrait des techniques d'estimation de taille à proprement parler (qui sont présentées dans les sections suivantes) afin de bien mettre en évidence les différences fondamentales avec la version de base du *CHS*[3].

Nous proposons de présenter ces modifications en suivant le fil d'une de ses itérations. Supposons donc que l'algorithme soit dans sa i^{e} itération, dans laquelle il doit localiser la cible se trouvant dans l'image numéro i de la séquence. La figure 5.4 permet de faciliter la compréhension des paragraphes suivants.

Dans une première phase la cible est d'abord localisée grossièrement avec une fenêtre de recherche et un modèle adaptés sur base du facteur d'échelle estimé à l'itération précédente r'_{i-1}. Si l'on suppose que la taille n'a pas *radicalement* changé de l'image *i-1* à l'image i et que *le facteur d'échelle a correctement été estimé à l'itération précédente*, la tolérance de la mesure d'intersection aux légères variations de taille devrait permettre au *CHS* de positionner sa fenêtre solution sur la région de la cible. Détaillons plus largement les hypothèses sûr lesquelles repose cette extension.

La première hypothèse fondamentale sur laquelle repose le *rCHS* est que le facteur d'échelle entre l'image de la cible à l'itération précédente et l'image de référence ait été estimé à sa véritable valeur (r_{i-1}). La valeur donnée par cette estimation r'_{i-1} sert de redimensionnement de "base" permettant de grossièrement remettre l'histogramme de la référence à l'échelle de celle de la cible, pour pouvoir effectuer une première localisation (sol'_i)

Cette hypothèse de type itérative implique qu'une estimation erronée du facteur d'échelle à une certaine itération, puisse conduire l'algorithme du *rCHS* à s'écarter définitivement de la bonne solution[4] pour les itérations suivantes. La dégénerescence est presque certaine en cas de perte momentanée de la cible ou de grave sous-estimation. Nous avons pu observer ce phénomène lors de certaines expériences.

La valeur initiale du facteur d'échelle doit être fournie à l'initialisation de l'algorithme. Celle-ci sera soit déterminée manuellement, soit par une procédure automatique (autre que le *CHS*[5])

[3]Ce qui nous permet par la même occasion de définir un cadre plus large de prise en charge des varations de taille sans être liés à une technnique d'estimation particulière.

[4]Nous entendons par "bonne solution" une localisation correcte de la cible.

[5]Cfr. échec de l'analyse d'intervalle [1].

La seconde hypothèse sur laquelle repose le *rCHS* est que la taille ne change pas de façon radicale d'une itération à la suivante. Celle-ci n'est toutefois pas trop contraignante dans la mesure où la cadence d'acquisition et de traitement est suffisament élevée que pour ne pas laisser le temps à la distance cible/caméra de varier de façon trop importante entre deux itérations successives.

Le modèle décrit par l'équation 5.4 fait intervenir un facteur d'échelle absolu r_i, calculé par rapport à une référence fixée. Une variante d'expression de ce modèle faisant intervenir un facteur d'échelle relatif, caractérisant la variation de taille par rapport à l'image précédente, plutôt qu'un rapport absolu nous semble mieux adaptée dans un contexte itératif où la solution donnée à une itération de l'algorithme dépend de la précédente. Ainsi, nous définissons v_i comme un facteur caractérisant la variation de taille de la cible entre l'itération *i-1* et l'itération i :

$$r_i = v_i r_{i-1} \tag{5.5}$$

Par ailleurs, cette formulation nous permet d'exprimer plus facilement la contrainte sur la dynamique de la composante du mouvement cible/caméra le long de l'axe optique de la caméra. La vitesse maximale à laquelle on veut que l'objet puisse se rapprocher ou s'éloigner de la caméra détermine, avec la fréquence d'acquisition et de traitement, une variation maximale de la taille de la cible, entre une image et la suivante. Cette limite sera modélisée à l'aide d'un facteur multiplicatif $v^{max} > 1$.

Dès lors, v_i est contraint d'appartenir à l'intervalle $[\frac{1}{v^{max}}; v^{max}]$. Donc, si la taille de la cible à l'itération $i-1$ vaut r_{i-1}, alors le facteur d'echelle r_i à l'itération i est contraint d'appertenir à l'intervalle $[\frac{r_{i-1}}{v^{max}}; v^{max} r_{i-1}]$ (figure 5.3)

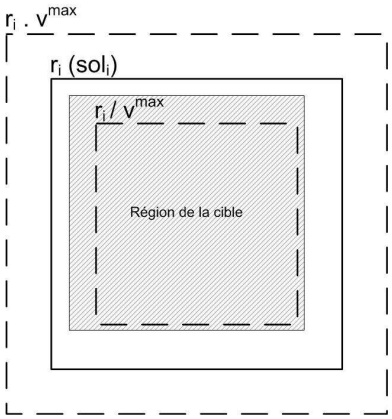


FIG. 5.3 – Hachuré : région cible + fenetre

Dans une deuxième phase, le facteur d'échelle absolu r_i (ou relatif v_i, l'un se déduisant de l'autre) est déterminé.

Pour cela, la fenêtre solution sol_i positionnée par le *CHS* est étendue par une bordure de b pixels, pour former ce que nous appelons la **fenêtre candidate**, qui est ensuite analysé par un algorithme d'extraction de taille(dont nous discutons dans

les sections suivantes) pour calculer le facteur d'échelle recherché.

Les algorithmes d'extraction ne peuvent fonctionner que sous une double conditions. Il faut que la fenêtre solution sol_i résultant de la première localisation (grossière) et la taille de la bordure ajoutée à cette fenêtre solution soient telles quel la fenêtre candidate résultante recouvre entièrement la région de la cible. Si celle-ci est trop petite ou qu'elle est placée à côté, l'extraction de taille échouera plus que probablement.

La valeur de cette estimation r_i' est finalement utilisée pour :

1. corriger la fenêtre sol_i résultant de la première localisation. Sa taille est réadaptée en fonction du facteur estimé, ce qui permet, pour autant que l'estimation soit fiable, d'augmenter la précision de la fenêtre solution définitive sol_i',
2. servir de facteur d'échelle de base à la localisation de la cible dans l'image i+1.

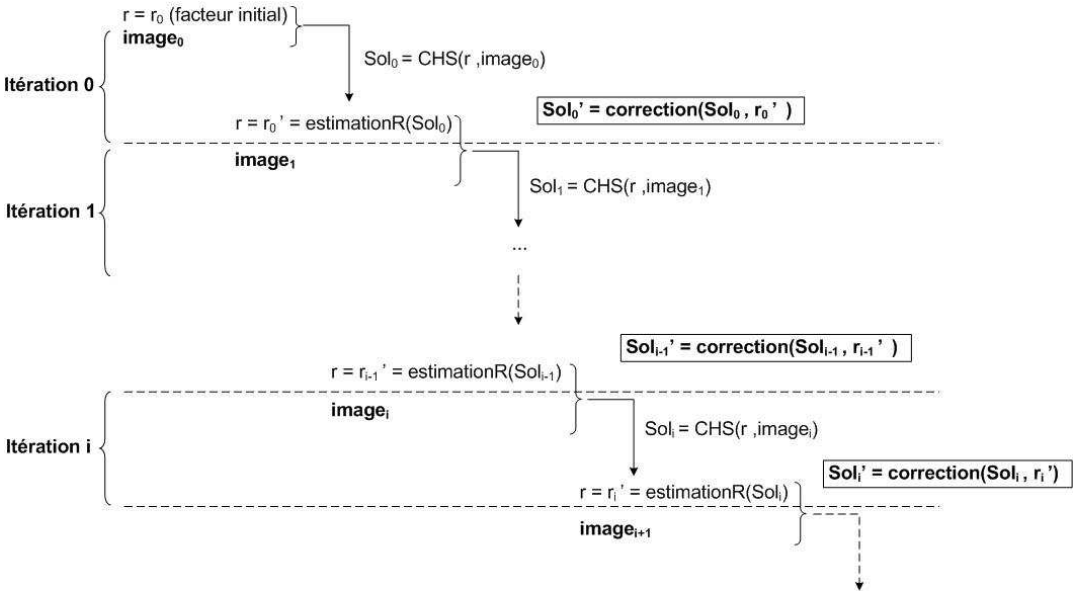


FIG. 5.4 – Le facteur d'échelle est utilisé d'une part pour affiner la solution à une itération i, et d'autre part comme base d'estimation servant pour la localisation de base de l'itération suivante $i + 1$. EstimationR est une fonction calculant le facteur d'échelle sur base de la fenêtre candidate construite autour de sol_i.

5.4 Estimation de taille par la méthode du *CHF*

5.4.1 Introduction

L'algorithme du ***CHF*** (**C**olour **H**istogram **F**ootprint) est un cas particulier du *rCHS*. Le schéma général ayant déjà été présenté dans la section précédente, nous nous concentrons ici sur le calcul de la taille à proprement parler.

Dans cette approche, la fenêtre candidate est d'abord filtrée par une fonction spécifique faisant apparaître sa silouhette dans une image binaire, nomée empreinte. Cette dernière est ensuite analysée à l'aide d'un algorithme de type bouding-box permettant de délimiter les contours de la cible pour finalement calculer sa surface, et donc déterminer le facteur d'échelle recherché.

Introduite dans [9] et brièvement développée dans [1], cette méthode semble fournir des résultats prometteurs qui encouragent la continuité du développement. Nous présentons d'abord le processus de formation et d'analyse de l'empreinte pour ensuite l'examiner en profondeur.

5.4.2 Principe

Supposons que l'algorithme du *CHF* soit dans sa i^{e} itération et que la fenêtre candidacte fc_i vienne d'être construite.

Formation de l'empreinte

L'idée de base est de faire apparaître dans une image binaire, dénomée empreinte, l'ensemble des pixels de la fenêtre candidate fc_i qui appartienent presque certainement à la région de la cible. Le nom d'empreinte vient du fait que cette image mette en évidence la silhouette de la cible (figure 5.4).

A cette fin, la fenêtre candidate est filtrée[6] à l'aide d'une fonction spécifique FP_α qui associe à chaque pixel la valeur 1 s'il appartient presque certainement à la la cible, la valeur nulle sinon.

Cette décision est prise en examinant la contribution d'un pixel couleur à la mesure d'intersection entre l'histogramme de la référence approximativement remis à l'échelle et celui de la fenêtre candidate. La définition initiale de la fonction FP_α est la suivante :

$$fc_i \mapsto FP_\alpha(fc_i) = emp \text{ où } emp(q) = \begin{cases} 1 & \text{si } \sum_{j=1}^{P} min(H_{fc_i}(j), \alpha\ r_{i-1}^2 H_{ref}\ (i)) \\ & \neq \sum_{j=1}^{P} min(H_{fc_i}^{\boldsymbol{-q}}(i), \alpha\ r_{i-1}^2 H_{ref}\ (i)) \\ 0 & \text{sinon} \end{cases} \quad (5.6)$$

où

[6] L'opération de filtrage consiste en la transformation par une fonction d'une image en une autre. Exemple : conversion d'une image en 256 niveaux de gris en une images en 32 niveaux de gris.

- fc_i est l'image délimitée par la fenêtre candidate,
- emp est l'empreinte de cette fenêtre,
- H_{fc_i} est l'histogramme de la fenêtre candidate, et $H_{fc_i}^{-q}$ le même histogramme duquel on a retiré un pixel de la classe q,
- H_{ref} est l'histogramme de l'image de référence,
- $\alpha \in [\frac{1}{v^{max}}; v^{max}]$ est un facteur permettant de sélectionner les pixels conservés dans la fenêtre empreinte (sa valeur est discutée sans la section 5.4.3),
- r_{i-1} est le facteur d'échelle estimé à l'itération précédente.

Cette définition peut se réécrire de façon équivalente, mais de façon plus simple sous la forme :

$fc_i \mapsto FP_\alpha(fc_i) = emp$ où

$$emp(q) = \begin{cases} 1 & \text{si } H_{fc_i}(j_q) \leq \alpha r^2 \ H_{ref}\ (j_q) \\ 0 & \text{sinon} \end{cases} \quad (5.7)$$

où j_q est l'index couleur du pixel q.

L'avantage de cette formulation est qu'elle peut immédiatement se transposer dans un algorithme de calcul de complexité linéaire par rapport à la taille de la fenêtre candidate.

Signalons que l'empreinte, qui est une image binaire, peut se prêter à des traitements morphologiques permettant d'en améliorer sa qualité pour, dans certains cas, faciliter l'extraction de taille.

Détermination de la taille

Un algorithme de type bounding-box est ensuite appliqué au contenu de la fenêtre empreinte afin de déterminer le contour de la cible. Dans sa version originale, le MIPS utilise un algorithme ne conservant que les lignes/colonnes contenant une proportion supérieure ou égale à 30% de pixels positionnés à un.

Combinaison des opérations

Le processus peut donc se résumer en 5 grandes étapes qui sont illustrées ci-après (figure 5.5).

- Première localisation avec le facteur d'échelle déterminé à l'itération précédente.
- Construction d'une fenêtre candidate par ajout d'une bordure à la fenêtre solution.
- Filtrage de cette fenêtre avec la fonction FP_α.
- Détermination des bords de la cible (opérations morphologiques et bounding-box).
- Correction sur la première localisation (pour donner plus de précision à la localisation d'une part, et pour la réutilisation dans les localisations ultérieures d'autre part).

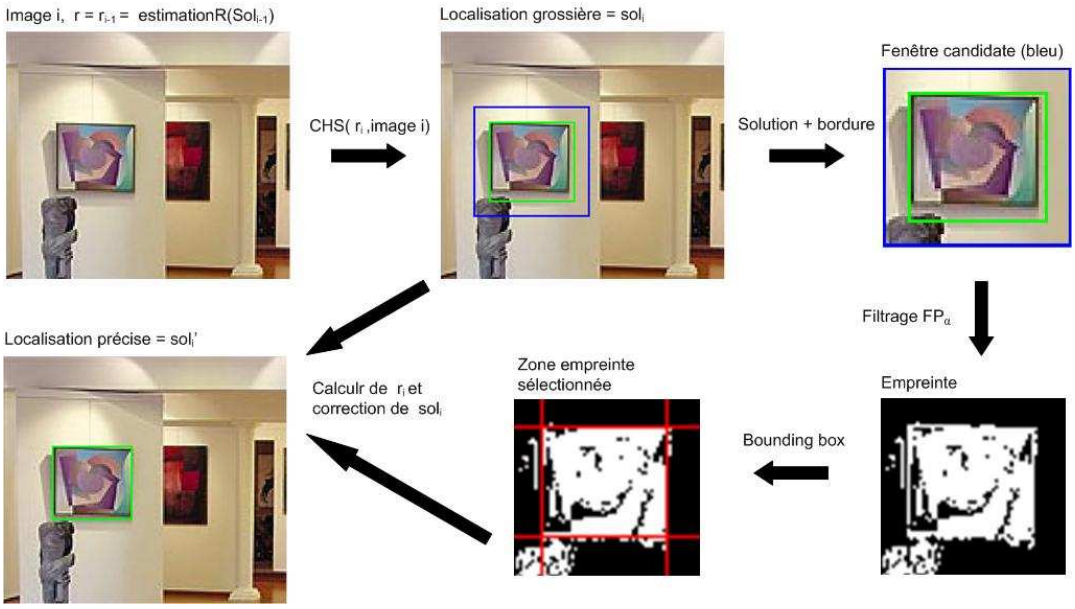


FIG. 5.5 – *CHF* : localisation du tableau le plus à gauche (vert = solution *CHS*, bleu = fenêtre candidate)

5.4.3 Analyse du *CHF*

L'utilisation de la méthode du *CHF* pose principalement deux problèmes. Le premier concerne la détermination du paramètre α. Le second concerne l'interprétation du contenu de la fenêtre empreinte.

Paramétrage

Le paramètre α joue un rôle critique dans le processus de formation de l'empreinte. Nous proposons de mener une étude de cas sur sa valeur par rapport à celle de la variation réelle de taille v_i, afin de déterminer s'il est possible d'en trouver une valeur optimale.

Considérons d'abord le cas de figure où la taille apparente augmente et étudions le comportement de la fonction FP_α en considérant respectivement les cas où $\alpha < v_i$, $\alpha = v_i$, $\alpha > v_i$.

Cas 1 : $1 < \alpha < v_i$. L'objectif de l'empreinte est de ne garder de la fenêtre candidate que les pixels qui contribuent certainement à la composition de la cible. Cette décision est prise à l'aide de l'équation 5.6 (ou 5.7). Un pixel q est conservé si l'inégalité est vérifiée, exclu sinon. Dans le cas où $\alpha < v_i$, les membres de droite ne sont jamais suffisamment élevés pour que les inégalités relatives aux différent pixels soient vérifiées. Il en résulte la suppression de la totalité (ou quasi-totalité) des classes de couleurs. La très grande majorité des pixels associés est donc par voie de conséquence mise à zéro, ce qui conduit l'algorithme d'extraction de taille à renvoyer une estimation de taille nulle. Le paramètre α doit donc être au moins aussi grand que l'augmentation réelle de taille.

Cas 2 : $\alpha = v_i$. Même si α égale exactement la véritable augmentation de taille apparente v_i, des effets non désirés peuvent encore apparaîtrent. En effet, le fait que des couleurs présentes dans la cible soient également présentes dans la bordure (ce qui n'est pas à exclure) implique que les inégalités correspondant à ces classes de couleurs ne sont pas vérifiées, et mettent donc à zéro les valeurs des pixels dans l'empreinte. Ceci entraîne donc une suppression de tous les pixels des classes correspondantes dans la fenêtre empreinte et mène de la même façon que précédemment à une sous-estimation. Sur des images réelles, le même problème de suppression "injustifiée" d'une classe de couleur peut également survenir en raison du bruit présent aux capteurs de la caméra.

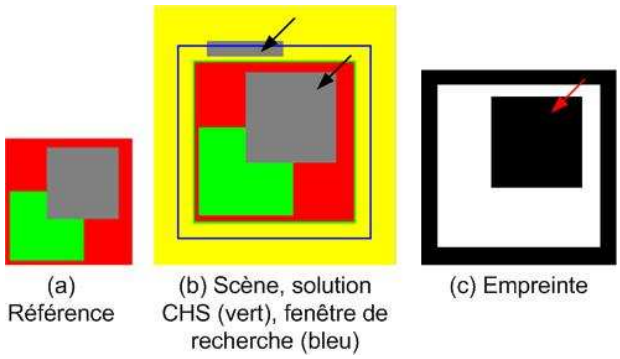


FIG. 5.6 – Image synthétique : classe grise supprimée à cause de la couleur grise présente dans la bordure.

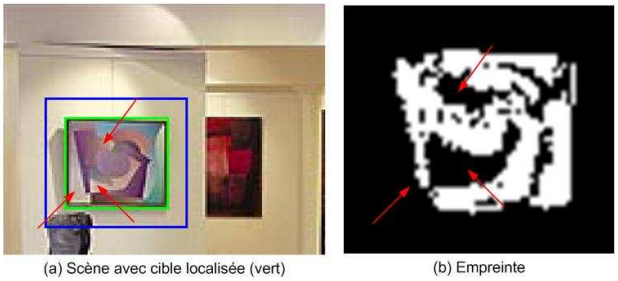


FIG. 5.7 – Image réelle : les classes de couleurs présentes à la fois dans le tableau et sur le mur.

Cas 3 : $\alpha > v_i$. Dans ce cas, le risque est que des couleurs, présentes dans le fond et en proportions différentes par rapport à celles qui sont dans la cible, soient injustement considérées comme faisant partie de la cible menant à une surestimation de la taille.

Le cas de figure dans lequel la taille apparente diminue (ie : $v_i < 1$) peut se ramener au dernier cas analysé. En effet, la similitude est immédiate lorsque l'on remarque que le paramètre $\alpha > 1$ sera toujours trop grand pour uniquement sélectionner les pixels d'une cible ayant subit une diminution de taille $v_i < 1$. La conséquence est aussi une surestimation probable du facteur d'échelle.

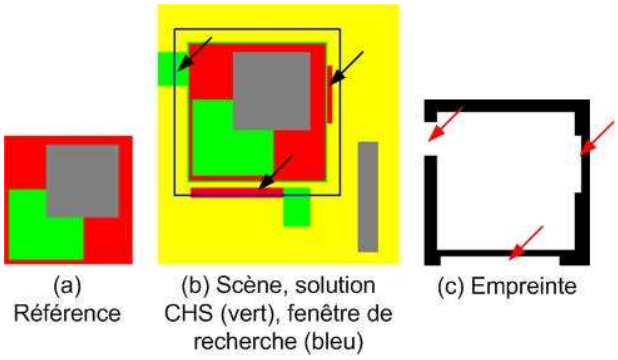


FIG. 5.8 – Image synthétique : classes de couleurs injustement considérées comme appartenant à la cible.

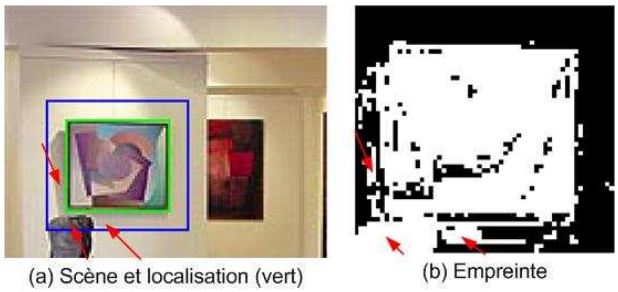


FIG. 5.9 – Image réelle : classes de couleurs injustement considérées comme appartenant à la cible.

Nous venons donc de démontrer qu'il n'est pas possible de trouver automatiquement une valeur optimale pour le paramètre α , pourtant critique dans la formation de l'empreinte. Une valeur trop petite de α conduit à une grave sous-estimation ou à un échec, un valeur trop grande induit un risque d'erreur élevé.

Ceci n'a en fait rien d'étonnant lorsque l'on se rend compte que la variation de taille et la distribution de couleur du fond sont a priori inconnues. Pratiquement, dans nos expériences nous avons fixé sa valeur à $v_{max}/2 \approx 1.15$, ce qui donnait d'assez bons résultats.

Interprétation de l'empreinte

Un second problème, survenant en aval de la formation de l'empreinte, est celui de l'interprétation de son contenu. En effet, bien que les contours puissent être facilement devinés par un observateur humain, il est toutefois difficile de mettre au point une procédure automatique d'extraction de la région de la cible pour en calculer la taille. Si l'empreinte est relativement bien formée, des opérations morphologiques (croissance de région, ...) peuvent y être appliquées pour la rendre plus facile à interpréter. Cependant, dans certains cas trop dégénérés, cette solution reste totalement inefficace. Les améliorations les plus intéressantes sont celles pouvant être apportées à la formation de l'empreinte.

5.4.4 Conclusion

Dans cette section, nous avons explicité l'algorithme du *CHF*, initialement introduit dans [9] permettant d'estimer le facteur d'échelle entre la cible et sa référence.

Nous avons ensuite mené une analyse théorique, appuyée par des images réelles et de synthèse, pour mettre en évidence deux problèmes majeurs.

Le premier étant l'intervention dans la fonction de calcul de l'empreinte d'un paramètre déterminant les pixels à garder dans l'empreinte, qu'il est impossible de fixer ou d'optimiser et dont dépend fortement la qualité de l'estimation. La raison de cette impossibilité étant que la variation de taille apparente d'une image à l'autre, la distribution de couleur du fond et la distribution du bruit des capteurs sont a priori inconnues.

Le second problème mis en évidence a trait à l'interprétation du contenu de l'empreinte après sa formation. En effet, même après traitement morphologique, nous avons vu que l'extraction de taille restait une opération complexe pas toujours possible à effectuer automatiquement bien qu'intuitivement évidente.

Les résultats donnés en pratique sont cependant relativement acceptables. Toutefois, en réponse à ces deux problèmes, nous proposerons dans la section suivante une technique d'estimation basée sur l'étude d'un indicateur.

5.5 Estimation de la taille par analyse sur famille d'indicateurs

5.5.1 Introduction

Dans cette section, nous présentons la technique d'estimation de la taille que nous avons développée par nous même durant notre stage et qui s'inscrit également dans le cadre général présenté en 5.3.

Celle-ci est basée sur une comparaison entre la forme de l'histogramme de référence et les différentes formes des histogrammes des sous-fenêtres de la fenêtre candidate. L'approche que nous développons ici fait abstraction de tout paramétrage critique et devrait être plus tolérante à l'environnement dans lequel évolue la cible. Nous commençons par en présenter le principe que nous formalisons ensuite afin de préciser les notions et établir quelques résultats appuyant l'intuition que nous avons de son fonctionnement. Nous analyserons ensuite quelques cas types, notamment ceux qui font échouer les autres méthodes, afin de vérifier le réel apport de notre contribution. Nous terminerons en mettant en évidence les points clefs marquant la différence par rapport aux autres solutions proposées par le passé.

5.5.2 Principe

L'approche que nous développons dans cette section est, tout comme la précédente, basée sur l'analyse de la fenêtre candidate.

Nous proposons d'exploiter une mesure de distance entre histogrammes afin de comparer des formes de distributions. L'idée est de localiser, dans la fenêtre de candidate, la plus grande sous-fenêtre dont la distribution de couleurs est la plus semblable possible à celle de la référence, au sens d'une certaine mesure. Cette sélection sera opérée par l'analyse des propriétés d'une famille d'indicateurs (tous fonction d'une variable[7] α) associés à des sous-fenêtres de la fenêtre candidate.

Notons par s_I toutes les sous-fenêtres possibles de la fenêtre de recherche. Appelons s_K la sous-fenêtre recouvrant exactement la région de la cible et notons α_K le facteur d'échelle y étant associé, le problème étant de déterminer la valeur de K.

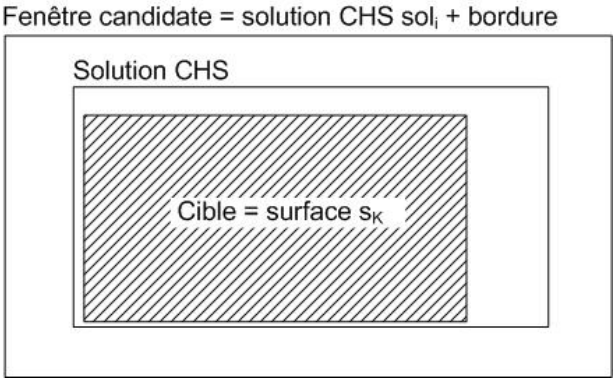


FIG. 5.10 – Sous-fenêtre de la fenêtre candidate recouvrant la région de la cible (hachurée).

Déterminer cette sous-fenêtre s_K revient à déterminer à la fois sa position et sa taille, ce qui constitue un problème à deux degrés de liberté. Nous proposons dans un premier temps de le simplifier en considérant que la cible est relativement bien centrée dans la fenêtre candidate, ce qui permet de fixer automatiquement la position de la fenêtre, éliminant ainsi un degré de liberté. Ce choix, pour des raisons de temps de calcul essentiellement, se justifie par le fait que si le mouvement de la cible et la fréquence d'acquisition/traitement sont tels que la taille apparente ne change que raisonnablement d'une itération à l'autre, le *CHS* doit être capable de positionner sa fenêtre solution sur la région de la cible. Ce fait a été vérifié expérimentalement.

Définissons donc une suite de sous-fenêtres imbriquées $(s_i)_i$, de taille strictement croissante et possédant les mêmes axes de symétrie que la fenêtre candidate. Associons-leur également leur facteur d'échelle α_i correspondant. Le problème revient donc à déterminer le plus grand indice k (c'est-à-dire la plus grande surface) qui est tel que l'histogramme de la surface s_k, que nous notons H_k, soit de même forme que celui de la référence. De façon plus formelle, nous recherchons la plus grande valeur de k telle que

[7] La variable α utilisée dans cette section n'a aucun lien avec le paramètre α de la fonction FP_α présentée dans la section précédente

$$H_k = \alpha_k \; H_{ref} \tag{5.8}$$

où α_k est le facteur d'échelle qu'il nous faut déterminer. En pratique, une égalité stricte n'est jamais obtenue en raison du fait que la cible n'est pas exactement centrée et que le signal délivré par la caméra comporte du bruit.

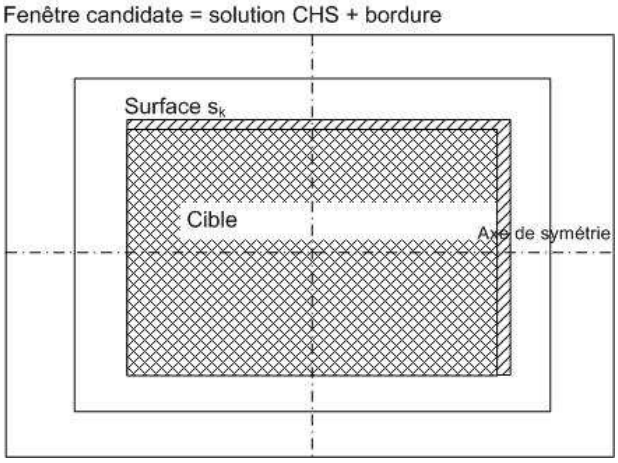


FIG. 5.11 – Sous-fenêtre centrée de la fenêtre de recherche

A chacune de ces sous-fenêtres s_i, associons un indicateur σ_i calculant la distance (norme L_1) entre l'histogramme H_i de la sous-fenêtre s_i et un multiple αH_{ref} de l'histogramme de la référence H_{ref} :

$$\sigma_i(\alpha) = \sum_{j=1}^{p} |\alpha \; H_{ref}\,(j) \; - H_i(j)| \tag{5.9}$$

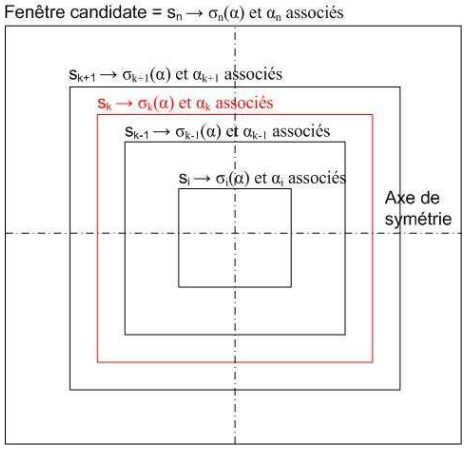


FIG. 5.12 – Fonction $\sigma_i(\alpha)$ associées aux sous-fenêtres

Montrons maintenant que l'étude des propriétés des indicateurs associés aux différentes sous-fenêtres de la fenêtre candidate permet de déterminer la valeur de k, et donc le facteur d'échelle recherché.

Pour $\boldsymbol{i = k}$, c'est-à-dire pour l'indice de la surface recouvrant la région de la cible, la fonction associée $\sigma_k(\alpha)$ admet un minimum global en $\alpha^* \approx \alpha_k$ (proposition 5.5.1).

Proposition 5.5.1 (Minimum global)

$$\sigma_k(\alpha) \textit{admet un minimum global en } \alpha^* \approx \alpha_k \tag{5.10}$$

Pour des valeurs de $\boldsymbol{i < k}$, $\sigma_i(\alpha)$ peut également présenter un minimum global représentatif d'une même forme de distribution, pour peu que la cible présente une certaine symétrie. Dans ces cas, certaines de ces valeurs minimales peuvent même être moins élevées que le minimum de la fonction σ_k, ceci étant dû au fait qu'il est toujours possible de trouver une "sous-cible" de même distribution perturbée par un bruit de plus faible densité. Ce phénomène ne nous est toutefois pas inconnu puisque c'est une des causes de mise en échec de la méthode par analyse d'intervalle [1].

Pour des valeurs de $\boldsymbol{i > k}$, les indicateurs $\sigma_i(\alpha)$ ne présentent jamais de minimum global et sont strictement croissants sur l'intervalle $[\alpha_{i-1}, \alpha_{i+1}]$ de variation de la variable α (proposition 5.5.2). Ceci repose sur l'hypothèse que la forme de l'histogramme identifie la cible dans la scène de manière unique et plus particulièrement qu'il n'existe pas de surface dans la scène ayant à la fois une forme de distribution de couleur et une taille supérieure à celle de la cible, ou encore :

$$\nexists k' > k \ tq \ H_{k'} = \alpha_{k'} H_{ref}.$$

Proposition 5.5.2 (Minimum global)

$$\forall k' > k, \sigma'_k(\alpha) \textit{n'admet pas de minimum global et est strictement croissante} \tag{5.11}$$

Ces propriétés nous fournissent un critère permettant de sélectionner la surface s_k où se situe la cible, le facteur d'échelle en découlant immédiatement : il nous suffit de déterminer la plus grande valeur de i telle que σ_i présente un minimum global.

Cette analyse peut, à première vue, sembler fastidieuse, mais il n'en est rien et ce pour deux raisons. En itérant de façon décroissante sur les valeurs de i, le nombre d'indicateurs à étudier est fortement limité. Deuxièmement, s'il existe un minimum global, nous avons vu que celui-ci se situerait à proximité du facteur d'échelle associé à la surface s_i, à savoir α_i. De plus, la plage de variation de α étant bornée inférieurement et supérieurement par α_{i-1} et α_{i+1}, l'intervalle sur lequel nous devons itérer pour déterminer la présence (ou l'absence) d'un minimum est fortement restreint.

Globalement la méthode donne d'assez bons résultats mais le décentrage de la fenêtre biaise parfois la solution. Il conviendrait d'étendre le critère de façon à déterminer simultanément position et taille. Nous ne nous y attarderons cependant pas

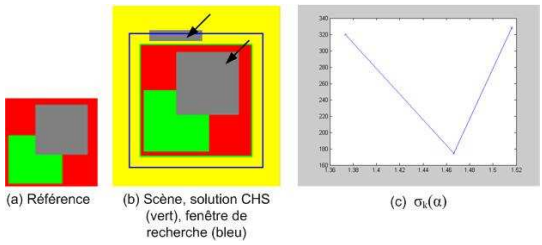


FIG. 5.13 – Estimation par analyse sur famille d'indicateurs moins sensible au fond que le *CHF* (exemple 1).

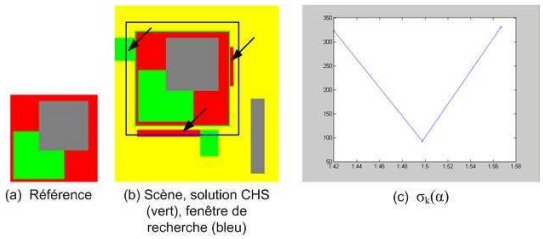


FIG. 5.14 – Estimation par analyse sur famille d'indicateurs moins sensible au fond que le *CHF* (exemple 2).

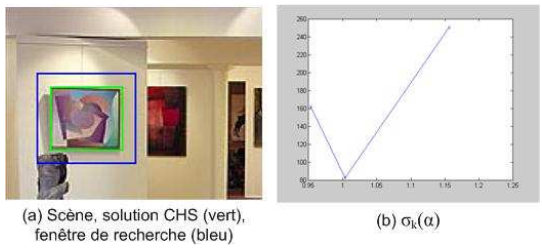


FIG. 5.15 – Estimation par analyse sur famille d'indicateurs sur une scène réelle.

plus étant donné que nous avons commencé simultanément le développement d'une autre technique permettant d'estimer position, taille et *orientation*.

La différence fondamentale avec l'analyse d'intervalle réside dans le fait que nous n'avons plus à rechercher un maximum local sur une courbe chaotique, s'étendant sur un grand intervalle, mais qu'à la place, il nous suffit de trancher sur l'existence ou la non-existence d'un minimum global, dont nous connaissons la position approximative lorsque celui-ci existe.

L'amélioration apportée par rapport au *CHF* est double. D'une part, la technique ne comporte aucun paramètre critique impossible à fixer. D'autre part, l'estimation de la taille est moins sensible au bruit et aux couleurs présentes dans l'environnement de la cible.

Cette approche ne permet toutefois pas non plus de déterminer si le facteur d'échelle est en sous-estimation, cas de figure qui contient à la dégénérescence de la solution.

5.5.3 Conclusion

Dans cette section nous avons développé une nouvelle technique d'estimation de la taille basée sur la seule information pertinente disponible dans le modèle, à savoir la **forme** de l'histogramme. En générant et analysant une famille d'indicateurs, chacune en correspondance avec une sous-fenêtre, nous avons montré qu'il était possible de déterminer la région de la fenêtre candidate recouvrant la cible et ainsi estimer sa taille. Afin de simplifier le problème, nous avons fait l'hypothèse que la cible était relativement bien centrée au sein de la fenêtre candidate. Il conviendrait d'essayer de généraliser la méthode pour éliminer cette condition et estimer simultanément taille et position. De plus, cette technique présente l'avantage d'être moins perturbée par l'environnement de la cible d'une part, et de ne plus comporter de paramètre critique d'autre part. Son bon fonctionnement a été validée lors des tests de suivis sur prototype robotique réel en fin de stage (chapitre 7).

5.6 Conclusion

Tout au long de ce chapitre, nous avons mis en évidence l'intérêt d'un ajustement dynamique de la taille.

Celle-ci est nécessaire pour assurer le suivi des mouvements faisant varier la distance séparant la cible de la caméra, induisant dans les images une variation de taille apparente pouvant être conséquente, excédant la tolérance de l'algorithme de base du *CHS*. D'autre part, une localisation à la fois fiable et précise ne peut être assurée qu'en connaissance du facteur d'échelle exact.

En outre et indépendamment de ces considérations, la taille est une information corrélée avec la distance séparant la cible de la caméra, et qui peut être utile dans le cadre d'asservissement visuel, notamment pour des applications de saisie semi-automatique d'objets dans lequel il faut diriger l'effecteur d'un bras robotisé juste devant l'objet cible.

Nous avons donc exposé une extension générale de l'algorithme du *CHS* ($rCHS$) prenant explicitement en compte les variations de taille. Cette approche modulaire nous a permis de bien mettre en évidence les différences fondamentales entre l'algorithme de base et l'algorithme étendu, la localisation y est effectuée en deux phases à la place d'une seule. Dans une première phase, la cible est d'abord localisée de façon approximative en se basant sur la taille déterminée à l'itération précédente. Ceci reposant sur la double hypothèse que la taille ait bien été estimée précédemment et qu'elle n'a pas radicalement changé depuis la dernière itération. La solution trouvée par le *CHS* est ensuite corrigée dans une seconde phase à l'aide d'une estimation du facteur d'échelle.

Cette estimation peut être réalisée par différents algorithmes dont nous avons ensuite discuté.

Comme proposé dans [1], nous avons étudié de manière plus approfondie la méthode du *CHF*, initialement introduite par Buessler et Urban dans [9]. L'expérimenta-

tion montre qu'elle donne de bons résultats dans beaucoup de situation. Cependant, l'étude de cas que nous avons réalisé montre qu'il est impossible de déterminer la valeur du paramètre intervenant dans la fonction de calcul de l'empreinte sur laquelle repose toute la méthode. De plus, dans certains cas le contenu de la fenêtre empreinte se révèle difficile ou impossible à interpréter.

Face à ces problèmes, nous avons tenté de développer une solution faisant abstraction de toute paramétrage. Celle-ci permet de sélectionner la région de la cible au sein de la fenêtre candidate en étudiant les propriétés d'une famille d'indicateurs liées à des ses sous-fenêtre. Notre approche repose actuellement sur une hypothèse de travail (fenêtre candidate centrée sur la région de la cible) qu'il conviendrait de lever en généralisant la méthode pour qu'elle considère toutes les sous-fenêtres possibles. Dans son état actuel, celle-ci fournit de bons résultats et se montre moins sensible au contexte dans lequel évolue la cible que la méthode du *CHF*. Cette méthode d'estimation a été utilisée dans l'expérience de validation sur prototype robotique réalisée en fin de stage (chapitre 7)

Ces deux techniques fonctionnent relativement bien pour l'estimation de taille de cible rectangulaires compactes[8] pour lesquelles le paramètre d'orientation peut être négligé. Envisager leur généralisation au cas des formes quelconques ne pourrait se faire sans prendre en compte ce paramètre, ce qui aurait pour conséquence d'introduire un degré de liberté supplémentaire dans le problème, augmentant par la même occasion le coût calculatoire des algorithmes.

Dans le chapitre suivant, nous présentons une approche originale basée sur une exploitation quelque peu différente du *CHS* permettant d'assurer le suivi de cibles tout en estimant simultanément leur taille et leur orientation.

[8] Pas trop allongées

Chapitre 6

Forme et orientation des cibles

6.1 Introduction

Dans les chapitres précédents, nous avons abordé de manière détaillée les problèmes de tailles et de conditions d'éclairage. Ces problèmes sont deux des principales difficultés auxquelles sont confrontés tous les chercheurs dans le domaine de la reconnaissance d'objets. Il existe cependant d'autres problèmes spécifiques à ce domaine, notamment celui de la forme de l'objet et de son orientation.

La performance des algorithmes que nous avons étudié dépend également de la forme et de l'orientation des objets dans l'image. Comme pour l'éclairage, les expériences montrent qu'il est possible de localiser de nombreux objets avec une inclinaison plus ou moins grande. Pour mieux comprendre cette tolérance, améliorer la robustesse de l'algorithme et pouvoir l'appliquer à des objets quelconques nous avons analysé les limitations de la technique. Ce chapitre présente des solutions que nous avons élaborées et testées...

En ce qui concerne la forme des objets, l'algorithme du CHS que nous utilisons balaie l'image au moyen d'une fenêtre de recherche rectangulaire. De même, lors de la phase d'estimation de la taille, la méthode de nos prédécesseurs se basait sur un certain taux de pixels, présents ou non, sur une ligne ou une colonne pour déterminer les frontières de l'objet et par conséquent sa taille. Dans la méthode d'estimation de la taille que nous avons proposée, nous nous basons aussi sur des fenêtres rectangulaires.

Ainsi, nous constatons que, dans le processus de localisation comme dans celui d'estimation de la taille, nous avons pris comme hypothèse que l'objet est rectangulaire. Or, lors de l'élaboration de notre problématique, nous avons insisté sur le fait que l'objectif du laboratoire TROP était de créer une technique de suivi d'objets en temps réel la plus universelle possible.

Nous souhaiterions, par conséquent, relacher cette hypothèse. Nos processus sont-ils encore utilisables ou, du moins, facilement adaptables à la recherche d'objets non rectangulaires ?

D'autre part, nous n'avons pas encore parlé de l'orientation de la cible par rap-

port à la caméra. Dans notre état de l'art, nous avons mentionné que la signature par histogramme est invariante à l'orientation. Mais la signature par histogramme ne constitue qu'une des nombreuses phases nécessaires à notre processus de suivi. L'invariance des histogrammes suffit-elle pour que notre processus de suivi soit tolérant aux changements d'orientations ?

Au cours de ce chapitre, nous allons essayer de répondre à ces questions, de voir si notre processus de suivi est généralisable aux objets quelconques et apte à tenir compte des changements d'orientation. Nous envisagerons enfin un autre processus de suivi, toujours basé sur la méthode du CHS, qui nous semble prometteur et qui nous permet d'estimer à la fois la taille et l'orientation de l'objet cible.

Nous avons décidé de rassembler ces deux problèmesdans un seul chapitre, même s'ils sont différents, car nous allons proposer une approche unique pour résoudre ces deux problèmes.

6.2 Les objets quelconques

Comme nous l'avons rappelé dans l'introduction, l'ensemble du processus de suivi d'objets que nous utilisons est basé sur des méthodes utilisant des fenêtres de recherche rectangulaires. Nous pouvons citer principalement la méthode du CHS, utilisée pour détecter un objet cible dans une scène, ainsi que les méthodes pour estimer la taille de l'objet dans cette scène. Ces méthodes sont donc optimales pour des cibles rectangulaires dont l'orientation dans l'image correspond à l'orientation de la référence.

Si de nombreux objets sont rectangulaires, il semble néanmoins évident que ceux-ci ne constituent qu'une infime partie des objets que nous pourrions être amenés à rechercher. Dans le cadre de la problématique initiale, c'est-à-dire l'assistance aux personnes handicapées au moyen d'un bras robotisé, le processus devrait être à même de guider le bras vers une tasse ou une bouteille d'eau, par exemple.

Dans quelle mesure notre processus de suivi est-il encore applicable ou, du moins, adaptable au suivi d'objets de forme quelconque ? Nous allons envisager plusieurs solutions pour répondre à cette question.

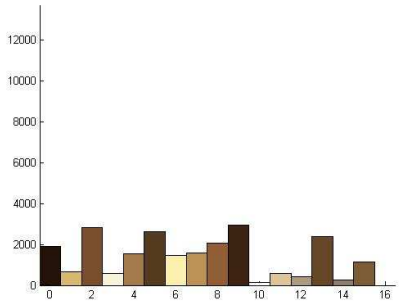
La première approche envisagée consiste à traiter les objets quelconques comme des rectangles en utilisant le rectangle circonscrit à l'objet en question. Nous envisagerons ensuite de localiser l'objet pour lui-même, c'est-à-dire d'adapter non plus l'objet à la méthode que nous avons validée pour les objets rectangulaires, mais bien d'adapter la méthode à l'objet cible. Dans ce cas, nous utiliserons un masque de l'objet recherché pour définir notre fenêtre de recherche. Enfin, nous proposerons une méthode hybride, alliant les avantages des deux premières méthodes.

6.2.1 Le cas des rectangles englobant les objets

Dans un premier temps, nous allons envisager que le fait de rechercher un objet quelconque peut être apparenté à rechercher la plus petite zone rectangulaire contenant cet objet. En d'autres mots, la thèse que nous essayons ici de valider est la suivante : "En prenant comme image de référence la plus petite zone rectangulaire dans laquelle l'ensemble de l'objet de référence est inscrite, nous obtenons une image de référence valide pour le suivi de cet objet." Si cette thèse s'avère exacte, le processus de suivi et les méthodes d'estimation de la taille ne nécessiteront pas d'adaptation.

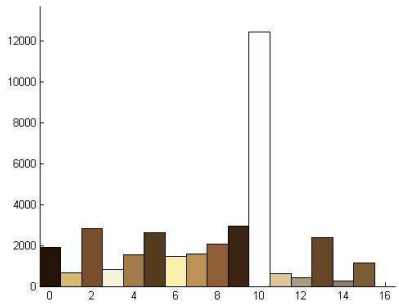
Cependant, cette hypothèse n'est pas valide. En effet, si l'on considère un objet quelconque, par exemple un sablier (figure 6.1), l'image de référence que nous allons utiliser pour localiser cet objet prendra en compte des pixels n'appartenant pas à l'objet, mais à son environnement. Dans le cas de notre sablier, nous avons environ 30% de pixels appartenant à l'environnement du sablier, pixels de couleur blanche pour la majeure partie.

(a) Objet de référence



(b) Histogramme de l'objet de référence

(c) Image de référence



(d) Histogramme de l'image de référence

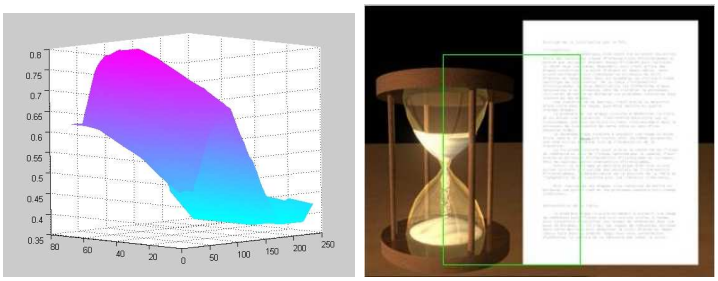
FIG. 6.1 – Le sablier, objet de référence, image de référence (prise sur fond blanc) et leurs histogrammes

Lors de la création de la signature de cette image de référence, c'est-à-dire son histogramme, chacun des pixels appartenant à l'environnement aura la même importance que n'importe quel pixel appartenant réellement à l'objet recherché.

Plus ces pixels sont nombreux dans l'image de référence, plus les risques d'erreurs lors de la localisation de l'objet par le CHS sont grands. En effet, si l'on suppose que x % des pixels de notre image de référence sont des pixels n'appartenant pas à l'objet de référence, en faisant abstraction du bruit des capteurs, nous savons que dans un autre environnement le maximum de similarité attendu pour l'objet en question serait compris dans l'intervalle $[0, 1 - x]$. Plus l'environnement de l'objet cible ressemblera à celui pris en compte dans l'image de référence, plus ce maximum de similarité se rapprochera de l'unité.

Un objet parasite, possédant les x % de pixels du fond de l'image de référence, possède au minimum une similarité de x % avec cette image et ce quelle que soit la couleur des autres pixels de cet objet parasite.

Dans le cas du sablier, notre image de référence inclut environ 30% de pixels n'appartenant pas à l'objet recherché. Nous constatons sur la figure 6.2 qu'une fenêtre de recherche comprenant ces 30% de pixels va être désignée comme étant la position de l'objet cible dans l'image, et ce, malgré sa faible ressemblance avec l'objet de référence. Par conséquent, notre processus de localisation a échoué et il est inutile d'envisager l'estimation de la taille par cette méthode.



(a) Surface de similarité

(b) Localisation erronée (rectangle vert)

FIG. 6.2 – Le sablier est mal localisé à cause de l'environnement, dans lequel une des pages de notre mémoire s'est glissée.

L'importance des pixels de l'objet de référence dépend directement du nombre de pixels n'appartenant pas à l'objet de référence, mais pris en compte dans la signature utilisée pour le retrouver. Plus ce nombre est élevé, plus cette technique devient inefficace et erronée.

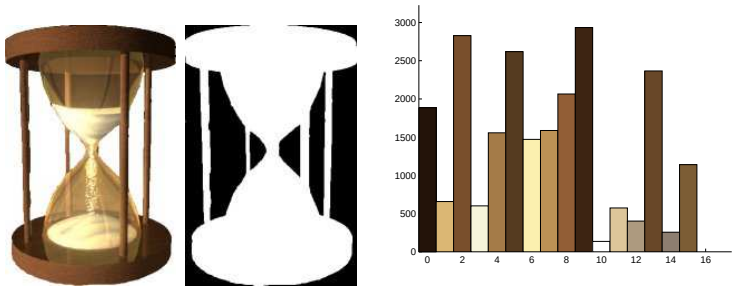
Bien que celle-ci possède l'énorme avantage de pouvoir utiliser nos algorithmes tels quels et ne nécessite par conséquent aucun traitement supplémentaire, nous sommes forcés de constater que si l'objet est de forme très découpée ou bien s'il présente des trous, cette approche risque de faire échouer la localisation.

6.2.2 L'utilisation d'un masque

Compte tenu du fait que nous faisons abstraction de l'orientation, la méthode la plus intuitive pour retrouver un objet est évidemment de faire évoluer une fenêtre de recherche de la forme de cet objet sur l'image. Cette méthode est la plus logique, celle qui nous vient directement à l'esprit pour rechercher un objet. C'est ce que nous faisions pour les cibles rectangulaires et il est tout naturel de vouloir réitérer ce procédé pour les objets quelconques.

L'utilisation d'un masque

Créer une varainte de l'algorithme pour chaque forme possible ne constitue évidement pas une solution envisageable. Afin de déterminer la forme de l'objet tout en gardant un procédé généralisable à n'importe quelle forme, nous allons utiliser un masque binaire. Ce masque est une image, de taille identique à celle de l'image de référence, dont chacun des pixels aura la valeur *un* si son homologue dans l'image de référence est un pixel de l'objet de référence, *zéro* dans les autres cas. La figure 6.3 illustre le masque associé à notre sablier ainsi que l'histogramme en résultant. Ce dernier est en fait un histogramme condditionnel. Seul les pixels de l'image de reférence dont l'homologue dans le masque a la valeur *un* sont pris en compte pour élaborer cet histogramme. Dans la figure 6.3, la couleur noire nous indique les pixels dont nous ne devons pas tenir compte tandis que la couleur blanche représente les pixels dont nous allons nous servir pour retrouver le sablier dans l'image cible.

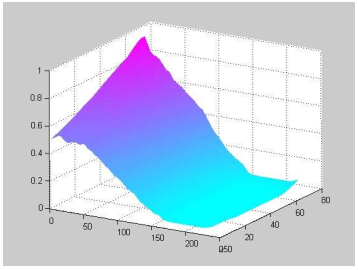


(a) Objet de référence (b) Masque (c) Histogramme de l'image de référence

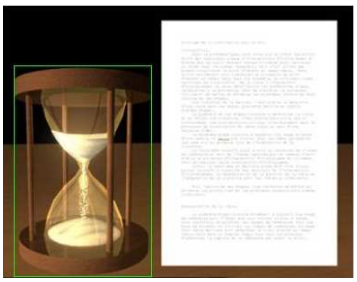
FIG. 6.3 – Le sablier de référence, son masque associé et son histogramme

L'introduction de masque dans le processus de suivi ne change pas fondamentalement le principe de ce processus. En effet, nous pouvons continuer à utiliser des fenêtres de recherches rectangulaires et utiliser le masque pour simuler une fenêtre de recherche adaptée à la forme et à l'orientation de l'objet. La seule différence par rapport à l'algorithme du CHS est qu'il s'agit, dans ce cas-ci, de faire un histogramme conditionnel de ces mêmes fenêtres de recherche et, donc, de ne prendre en compte que les pixels dont l'homologue dans le masque a la valeur *un*.
La figure 6.4 nous montre la position de la fenêtre de recherche correspondant au maximum de similarité. Afin de vérifier si cette fenêtre de recherche est celle que nous voulons, la figure 6.4 présente aussi l'objet cible correspondant à cette fenêtre

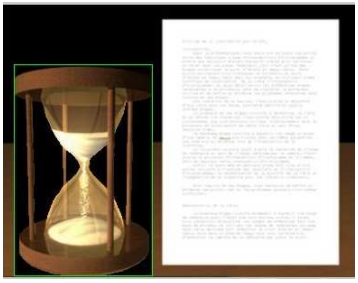
de recherche, que nous mettons en évidence en multipliant les pixels de la fenêtre de recherche par la valeur de ceux du masque. Ainsi, seuls les pixels pris en compte dans l'élaboration de l'histogramme sont affichés, les autres ont la valeur RGB [0 0 0] et sont par conséquent affichés en noir.



(a) Surface de similarité



(b) Localisation de la fenêtre de recherche correspondant au sablier (rectangle vert)



(c) Mise en évidence du sablier au moyen du masque

FIG. 6.4 – Utilisation d'un masque pour la localisation

Nous constatons que, visuellement, l'objet mis en évidence de cette manière correspond à l'objet cible ; la localisation semble donc exacte.
Afin d'estimer la taille de l'objet, nous allons procéder de la même manière que pour les cibles rectangulaires, mais en utilisant ici aussi des histogrammes conditionnels, tenant compte du masque.

Implémentation du masquage

Les modifications à apporter à l'algorithme du CHS pour prendre en compte des fenêtres de recherche quelconques, par le bied de l'utilisation de masques, ne sont pas très complexes.
Dans le cadre des cibles rectangulaires, l'algorithme intuitif de calcul du CHS consistait à générer l'histogramme de chaque fenêtre de recherche et de le comparer à celui de l'objet de référence. Pour les cibles quelconques, nous procéderons de manière identique. Il nous suffit de rajouter un test pour vérifier, dans le masque, si le pixel que l'on traite doit ou non apporter sa contribution à l'histogramme de la fenêtre de recherche.

Cependant, une telle implémentation n'est pas applicable au temps réel. Recalculer l'entièreté de l'histogramme pour chaque fenêtre de recherche implique un trop grand nombre d'opérations. En se basant sur la méthode de calcul optimisée que le laboratoire TROP a développée pour les fenêtres rectangulaires, la solution consisterait ici aussi à identifier les pixels qui contribuent à la fois à la fenêtre de recherche actuelle et à la fenêtre de recherche suivante, de manière à limiter le nombre de pixels à prendre en compte. Si nous effectuons la différence entre le masque à la position i et le masque à la position $i + 1$, nous mettons en évidence les pixels à ajouter et ceux à soustraire pour obtenir l'histogramme de la fenêtre de recherche suivante. La figure 6.5 illustre ce phénomène pour notre sablier. Les pixels bleus sont ceux dont la contribution devra être ajoutée pour la fenêtre de recherche suivante et les pixels rouges, ceux dont la contribution ne devra plus être prise en compte.

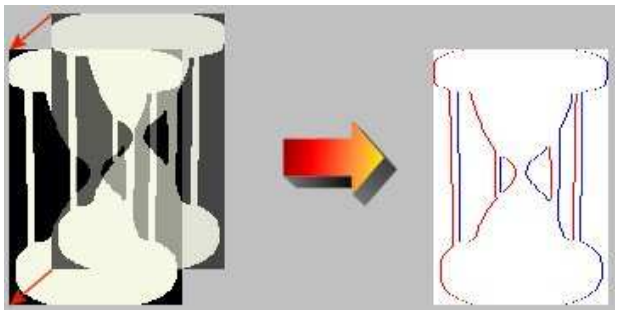


FIG. 6.5 – Pixels à prendre en compte dans la méthode optimisée

Si, dans le cas des cibles rectangulaires, il s'agissait d'une colonne à ajouter et d'une autre à soustraire ; dans le cas des formes quelconques, ces pixels ne forment plus nécessairement deux colonnes. Dès lors, il nous est nécessaire de disposer de la liste des positions relatives, par rapport au coin supérieur gauche de la fenêtre de recherche, des pixels à ajouter et à soustraire à l'histogramme courant. Afin d'obtenir l'histogramme de la fenêtre de recherche suivante, nous devons parcourir cette liste pour accéder aux pixels à prendre en considération. Il s'agit exactement du même procédé que celui que nous avons utilisé dans le cadre des cibles rectangulaires, à ceci près que celui-ci nécessite une opération supplémentaire pour accéder aux pixels dont la contribution à l'histogramme doit changer. D'après nos estimations, cette indirection nécessaire pour atteindre les pixels implique une perte de performance de l'ordre de 25%.

En outre, nous pourrions être amenés à prendre en compte beaucoup plus de pixels que dans le cas des cibles rectangulaires. Sur l'exemple du sablier, nous avons plus de deux fois plus de pixels à prendre en compte, principalement à cause des montants consolidant le sablier. Si nous voulons rechercher un peigne, la situation est encore bien pire. Pour chacune des dents de ce peigne, il nous faut retirer une rangée de pixels et en ajouter une autre. En comparaison avec une cible de même taille, il nous faudra tenir compte de presque n fois plus de pixels, où n représente le nombre de dents de ce peigne.

En ce qui concerne l'implémentation de l'estimation de la taille, le problème ma-

jeur va consister à adapter la taille du masque. En effet, pour chaque taille envisagée, l'histogramme doit être calculé. Il nous faut donc, a chaque itération du suivi, calculer plusieurs masques afin d'estimer la taille de l'objet dans la scène.

Conclusion

En ayant recours aux masques, la détection de cibles quelconques est aussi fiable qu'elle l'est pour les objets rectangulaires. L'utilisation de masques nous confère donc un algorithme adapté pour la détection de cibles quelconques. Cependant, cette méthode engendre des opérations supplémentaires qui peuvent devenir critiques. Nous allons donc envisager une méthode intermédiaire, qui nous permet de garder une détection fiable tout en limitant les calculs supplémentaires nécessaires à la prise en compte des objets non rectangulaires.

6.2.3 Solution hybride : l'histogramme propre

Afin de garder de bonnes performances, nous allons revenir aux fenêtres de recherches rectangulaires classiques, c'est-à-dire sans l'utilisation de masques, mais en nous efforçant d'éliminer le problème lié aux pixels du fond. En travaillant avec des fenêtres de recherche rectangulaires, nous nous retrouvons dans la situation où, pour passer de l'histogramme d'une fenêtre de recherche à celui de la fenêtre suivante, il suffit de retirer la contribution des pixels de la première colonne de la première fenêtre de recherche et d'y ajouter celle des pixels de la dernière colonne de la seconde fenêtre de recherche. Par la même occasion, cela nous évite les indirections nécessaires pour accéder aux pixels dont nous devons ajouter ou retrancher la contribution.

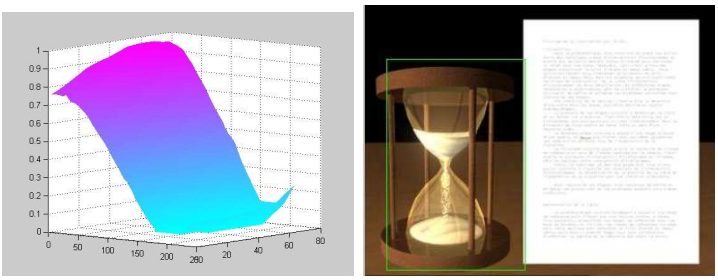
Le masque que nous avons introduit dans la section 6.2.2 sera encore utilisé dans cette technique, mais uniquement lors de l'élaboration de l'histogramme de l'objet de référence, afin de ne pas y comptabiliser les pixels n'appartenant pas à l'objet recherché. Nous nommerons cet histogramme l'histogramme propre de l'objet, car celui ne contient que les pixels propres à l'objet et il n'inclut pas, dans la signature, des pixels parasites entravant les performances du suivi.

Nous l'avons déjà mentionné dans le chapitre décrivant l'algorithme de suivi, nous utilisons, pour mesurer la similitude entre deux signatures, l'intersection d'histogrammes telle qu'elle a été définie par Swain et Ballard dans [29]. Cette définition nous permet de comparer les histogrammes des différentes fenêtres de recherche avec celui de l'objet, même si ces histogrammes ne comportent pas le même nombre de pixels.

Si nous avions utilisé une autre métrique pour comparer les histogrammes, par exemple une des métriques traditionnelles telles que les normes L_1 ou L_2 , le résultat aurait été différent. Si le fait de ne pas avoir une similarité maximale n'est pas un problème en soi, d'autres phénomènes liés à ces normes sont beaucoup plus gênants. La norme L_2 , par exemple, ne donnerait pas la même similarité à deux objets cibles identiques dans des fonds différents, car elle calcule la distance entre les deux histogrammes en mettant au carré la différence de pixels pour chacun des bins. En ef-

fet, ces métriques ne comparent pas le recouvrement entre deux histogrammes, mais plutôt leurs différences. L'emploi de la mesure de similarité telle que définie dans [29] nous assure que, dès qu'une fenêtre de recherche possédera les mêmes pixels de couleurs que ceux de l'objet de référence, elle aura une similarité maximale, et ce, quelle que soit la couleur des autres pixels présents dans cette fenêtre de recherche.

Cette méthode fournit une localisation moins précise que celles où nous faisons intervenir le masque. Sur la figure 6.6, nous constatons que l'utilisations de l'histogramme propre permet une meilleure localisation que la méthode présentée dans la section 6.2.1, sans pour autant avoir la précision de la localisation au moyen des masques. Le pic dans la surface de similarité est moins marqué, et un plateau dans la surface de similarité empêche l'algorithme de déterminer un maximum unique. Cependant, il est important de noter que le voisinage du sablier, dans la scène, possède de nombreuses couleurs également présente dans le sablier.

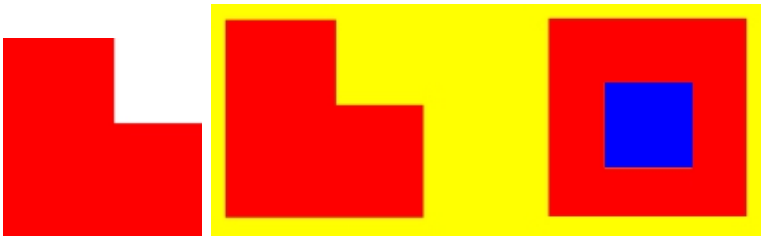


(a) Surface de similarité (b) Localisation de la fenêtre de recherche (rectangle vert)

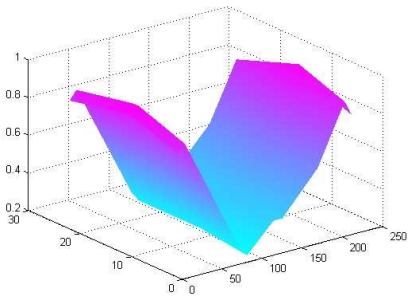
FIG. 6.6 – Localisation du sablier via l'histogramme propre

Cette méthode est moins précise que celle où nous faisons intervenir le masque dans chaque fenêtre de recherche. La figure 6.7 ilustre deux objets pour lesquels leurs fenêtres de recherche respectives ont une même valeur de similarité, bien qu'ils soient différents. Les fenêtres de recherche n'ont pas les mêmes histogrammes, mais, comme toutes les deux contiennent l'ensemble des pixels présents dans l'histogramme propre de l'objet recherché, leurs similarités sont identiques. La méthode utilisant les masque (section 6.2.2) ne souffre pas de cet inconvénient, car elle compare deux histogrammes contenant le même nombre de pixels. Dans ce cas, si la similarité est égale, les zones sont par conséquent de distribution de couleurs identiques. Cependant, l'utilisation des masques s'avère très coûteuse en calcul notamment pour l'adaptation de la taille et de l'orientation du masque.

Dans le second cas, nous constatons que cette méthode est aussi plus sensible à l'environnement. En effet, si dans l'environnement de l'objet, des couleurs de l'objet sont présentes, la localisation risque d'être imprécise. Plus la différence entre le nombre total de pixels de l'histogramme de référence et celui de l'histogramme de la fenêtre de recherche est grande, plus la localisation de l'objet de référence risque d'être imprécise. Le nombre de pixels de l'objet de référence, qui jouxtent les bords de la fenêtre de recherche associée, est aussi un facteur déterminant en ce qui concerne la localisation précise de l'objet cible. Sur la figure 6.8, nous constatons que pour un



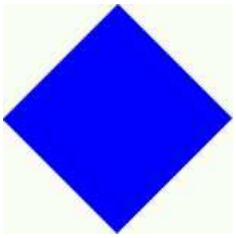
(a) Référence (b) Scène



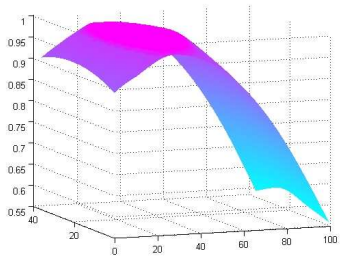
(c) La surface de similarité présente 2 maxima égaux

FIG. 6.7 – Deux objets dont les fenêtres de recherches sont différentes peuvent être confondus.

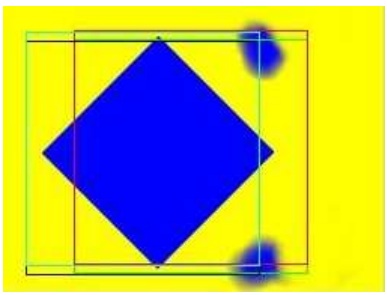
losange, seuls quelques pixels de l'objet de référence jouxtent les bords de la fenêtre de recherche associée. Si dans l'environnement de l'objet cible une zone de l'image possède des pixels identiques à ceux présents en bordure de l'image, plusieurs fenêtres de recherches auront vraisemblablement une similarité maximale. La localisation sera par conséquent moins précise.



(a) Référence



(b) Surface de similarité présentant un plateau



(c) Scène et quelques unes des fenêtres de recherches pour lesquelles la similarité est la plus grande

FIG. 6.8 – Plus le nombre de pixels jouxtant la fenêtre de recherche est faible et moins la détection est précise.

Cependant, il ne faut pas oublier que la seule hypothèse que nous avons prise afin d'utiliser le CHS est qu'aucun autre objet de la scène n'aura la même distribution de couleurs, hypothèse qui, comme nous l'avons précisé lors de la description du processus de localisation, n'est pas très restrictive pour des objets riches en couleurs.

Cette hypothèse n'est pas encore suffisante, car, dans notre cas, nous ne recherchons pas directement l'objet, mais bien une fenêtre de recherche contenant l'objet. Afin d'éviter que deux objets de formes différentes, mais ayant une distribution identique de couleurs, ne se retrouvent sur la même scène, nous allons quelque peu étendre cette hypothèse. Nous allons prendre comme hypothèse qu'aucune zone de l'image, d'une taille égale à celle de la fenêtre de recherche utilisée pour localiser l'objet, ne sera similaire à l'objet cible. Si nous nous limitons aux objets rectangulaires, cette hypothèse est strictement équivalente à celle que nous avions précédemment acceptée. Par contre, dans le cadre des objets quelconques, elle nous permet d'utiliser cette technique sans nous préoccuper de la forme des objets.

Nous avons maintenant un algorithme apte à la localisation d'objets quelconques aussi rapide que dans la version pour les formes rectangulaires, et ce, en n'impliquant que de légères modifications. Il nous reste à estimer la taille de l'objet.

Notre méthode d'estimation de taille est basée sur la comparaison de distribution de couleurs d'une image avec d'autres fenêtres de recherche. Afin d'effectuer la comparaison des distributions de couleurs, nous devons repasser par l'utilisation du masque car nous avons besoin d'une forme précise à faire évoluer pour obtenir une estimation précise de la taille.

6.2.4 Conclusion

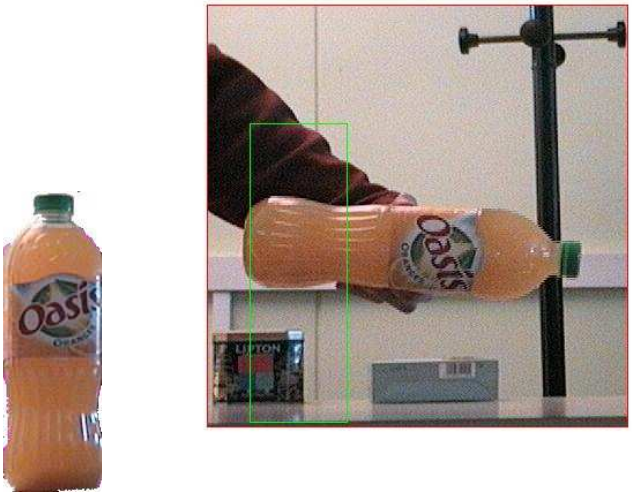
Nous avons pu établir une solution apte au temps réel qui nous permet de suivre des objets quelconques. Cette méthode n'implique aucun calcul supplémentaire, mais peut s'avérer moins précise qu'une méthode prenant en compte l'orientation de la cible. Nous avons aussi élaboré une autre métode qui est aussi fiable et précise que celle que nous utilisions pour les cibles rectangulaires, mais qui nécessite quelques calculs supplémentaires. Nous avons décidé d'utiliser celle qui n'en nécessite aucun. Cependant, si cette méthode ne nous suffit pas, nous pourrons toujours utiliser celle qui recourt aux masques.

6.3 L'orientation de la cible dans l'image : limites de tolérance du CHS

Nous allons maintenant aborder un autre problème, celui de l'orientation de la cible dans l'image. Rien ne nous permet d'affirmer que la cible aura la même orientation que l'objet de référence. Nous allons donc étudier les limites de tolérance du CHS face à ce problème, et enisager différentes approches pour essayer d'augmenter la tolérance du processus de suivi aux changements d'orientation.

Une des raisons pour lesquelles nous avons choisi les histogrammes, c'est qu'ils ont l'avantage d'être indépendants de l'orientation. En effet, les histogrammes ne tiennent aucun compte de la position des pixels dans l'image. Ils se cantonnent à dénombrer, pour chaque couleur présente dans l'image, les pixels de l'image de cette couleur. Par conséquent, si nous élaborons l'histogramme d'un objet, quelle que soit son orientation, l'histogramme reste rigoureusement identique.

Cependant, la méthode du CHS est basée sur le parcours de la scène avec une fenêtre de recherche afin de localiser un objet cible. Jusqu'à présent, la fenêtre de recherche que nous utilisions avait la même orientation que l'objet de référence. Si nous recherchons un objet, qu'il soit ou non rectangulaire, dont la fenêtre de recherche associée a un rapport hauteur-largeur proche de un, nous n'aurons aucun problème lors de la localisation de l'objet. En effet, dans ce cas, l'objet cible sera majoritairement contenu dans une des fenêtres de recherche et la localisation se déroulera sans problème. Si nous recherchons un objet dont le rapport hauteur-largeur est éloigné de un, dans le cas où l'orientation de la fenêtre de recherche ne correspond pas à celle de l'objet cible dans la scène, il nous sera impossible de le localiser précisément. Il est probable que la fenêtre de recherche recouvre partiellement l'objet cible, mais, même dans ce cas, cela ne nous suffit pas pour localiser l'objet précisément. Nous pouvons visualiser ce phénomène sur la figure 6.9.



(a) Référence (b) Localisation

FIG. 6.9 – Localisation en ne tenant pas compte de l'orientation.

Une première approche pour être sûr que la fenêtre de recherche recouvre l'objet cible serait d'augmenter la taille de la fenêtre de recherche. Supposons que l'on recherche un objet dont la fenêtre de recherche associée a une hauteur de m pixels et une largeur de n pixels, utiliser une fenêtre de recherche carrée de coté $\sqrt{m^2+n^2}$ nous permet d'affirmer que, quelle que soit l'orientation de la cible par rapport à la caméra, cette fenêtre de recherche sera à même de contenir l'objet cible. Cependant, une fenetre de recherche surdimensionnée ne permet pas de localiser avec précision l'objet. Cette méthode donne lieu à un plat dans la surface de similarité et il y a donc plusieurs positions valides. Chacun de ces maxima nous donne la position d'une fenêtre de recherche contenant l'objet, mais pas la position précise de cet objet au sein de cette fenêtre, ni son orientation. Nous ne pouvons donc pas nous contenter de cette méthode.

Puisque cette méthode ne nous permet pas une bonne détection, nous allons en-

visager d'estimer l'orientation et d'adapter la fenêtre de recherche en conséquence. Nous allons procéder comme nous l'avons fait pour l'estimation de la taille, c'est-à-dire que nous allons nous baser sur le résultat précédent afin de retrouver l'objet dans la scène et affiner l'orientation après chaque localisation. Nous supposerons, ici aussi, que les variations d'orientations d'une image à l'autre seront limitées. Après avoir localisé l'objet, il nous faudra déterminer avec précision l'orientation de l'objet dans la scène, afin de pouvoir le localiser à l'itération suivante.

Il s'agira tout d'abord de déterminer la variation d'angle maximum que nous accepterons entre deux captures successives. Ensuite, pour chaque variation d'orientation acceptée, nous calculerons l'histogramme correspondant et l'intersection de cet histogramme avec celui de l'objet de référence. L'orientation pour laquelle la valeur de similarité sera maximale sera considérée comme l'orientation réelle de l'objet.

Cependant, cette méthode n'est pas dépourvue d'inconvénients. L'implémentation de la rotation d'une fenetre de recherche revient à utiliser la technique des masques (présentée dans la section 6.2.2). Le coût est trop élevé car il nous faut recalculer le masque à utiliser pour chaque orientation possible. Par ailleurs, cette méthode n'a de sens que si nous avons la certitude que la fenêtre de recherche est centrée sur l'objet. Si tel n'est pas le cas, lors de l'essai des masques correspondant aux différentes orientations envisagées, ceux-ci ne seront pas centrés sur l'objet et aucun ne correspondra précisément à l'objet cible.

Enfin, nous avons un autre problème qui, lui, n'est pas propre à cette méthode, mais plutôt au fait de vouloir estimer la taille et l'orientation d'un objet. Entre deux captures successives, à la fois la taille et l'orientation sont susceptibles d'avoir évolué. Quel facteur devons-nous estimer en premier ?

En fait, ces deux facteurs devraient être évalués en même temps, ils sont interdépendants et cela nous pose problème. En effet, si l'orientation de la cible n'est pas exacte, nous ne saurons déterminer la taille avec précision, et vice versa. En répétant successivement les estimations de la taille et de l'orientation, il est possible que celles-ci convergent. Mais cela implique des suppléments de calculs à chaque itération. Nous nous sommes tournés vers une autre approche, qui nous semble plus prometteuse.

6.4 La recherche de motifs

6.4.1 Introduction

Au vu de l'insuffisance des méthodes proposées précédemment, nous nous sommes tournés vers une méthode inspirée des travaux de [20].Dans leurs travaux, [20] recherchent des cibles elliptiques dans des scènes en niveau de gris. L'originalité de leur méthode vient du fait, qu'au lieu de rechercher cet objet elliptique dans l'image, ils recherchent les quatre quartiers de l'ellipse indépendamment les uns des autres. A partir des positions relatives de ces quartiers, ils peuvent déduire l'orientation de l'objet dans la scène.

De manière similaire, nous allons tenter de déterminer la taille et l'orientation de l'objet cible au moyen de certaines zones particulières de celui-ci. Pour chaque objet de référence, nous allons déterminer des zones particulières, que nous nommerons des motifs. A partir des positions relatives de ces motifs dans la scène, nous allons estimer la taille et l'orientation de l'objet cible.
Dans cette section nous traiterons donc, dans un premier temps, des caractéristiques nécessaires à une zone de l'image pour constituer un bon motif et de l'existence de telles zones dans l'objet de référence. Nous allons envisager un algorithme de sélection de motifs basé sur les caractéristiques proposées. Ensuite, nous aborderons l'élaboration d'un algorithme de suivi apte à localiser les motifs. Enfin, nous allons brièvement présenter comment, à partir des positions des motifs, nous allons pouvoir retrouver la taille et l'orientation de l'objet dans la scène.

6.4.2 Les motifs

Après avoir acquis l'objet de référence et l'avoir indexé, nous allons déterminer les motifs que nous utiliserons pour suivre l'évolution de l'objet de référence par rapport à la caméra. Avant tout, il est nécessaire de clarifier ce que nous entendons par le terme "motif".

Un motif est une zone particulière de l'objet de référence dont la similarité avec n'importe quelle autre zone de l'objet de référence est la plus faible possible. En d'autres termes, un motif est une zone aisément identifiable par l'algorithme du CHS au sein de l'objet de référence.

Pouvons-nous affirmer que des motifs ainsi définis existeront au sein de tous nos objets de référence ? Premièrement, l'objet recherché constitue déjà un motif. En effet, aucun sous-élément de l'objet de référence n'aura une similarité plus grande avec cet objet. D'autre part, comme Andre et Frippiat l'ont mentionné dans [1], la détection d'objet par la méthode d'intersection d'histogrammes fournira de bons résultats si l'objet recherché est riche en couleurs. Dans le cas des objets riches en couleurs, nous pourrons naturellement trouver des zones qui constitueront de bons motifs.

Afin de pouvoir estimer la taille et l'orientation d'un objet cible, nous devons disposer d'un minimum de deux motifs. A partir de ces deux motifs, nous serons aptes à estimer la taille et l'orientation de l'objet cible dans la scène, du moins en ce qui concerne les rotations dans le plan orthogonal à l'axe de la caméra.

La manière la plus simple pour choisir des motifs est de les sélectionner manuellement, en prenant des motifs qui, visuellement, nous semblent intéressants. Cependant, nous avons envisagé une autre manière, plus rigoureuse, pour mettre en évidence les motifs les plus aptes à être suivis. Nous avons, dans un premier temps, élaboré quelques critères qui nous semblent pertinents pour choisir les motifs les plus prometteurs :

- Favoriser les motifs possédant des couleurs marqueurs. Les couleurs marqueurs sont des couleurs qui n'apparaissent que dans ce motif, et nulle part ailleurs

dans l'objet de référence. Ces couleurs nous assurent qu'aucune autre zone de l'image n'aura une similarité maximale avec ce motif.

- Favoriser les motifs riches en couleurs pour lesquels le CHS fournira de bons résultats.

- Imposer une taille minimale aux motifs, afin de limiter l'impact d'un pixel bruité et ainsi de limiter les erreurs de localisations dues au bruit.

- Favoriser les motifs pour lesquels la surface de similarité fournit un pic secondaire faible, c'est-à-dire les motifs ayant une distribution de couleurs très différentes des autres zones de l'objet de référence. Par pic secondaire nous entendons la plus haute valeur de la surface de similarité en retranchant à celle-ci l'influence des pixels du motif recherché.

Afin de rechercher de tels motifs, nous avons ensuite élaboré un algorithme qui sélectionne les motifs les plus aptes en fonction de ces critères. Nous avons donc envisagé une pondération de ces critères et recherché les motifs y correspondant. Les motifs élus par l'algorithme varient selon la pondération utilisée. Cependant, si nous prenons l'exemple d'une bouteille, nous constatons que, quelle que soit la pondération de ces critères, l'algorithme sélectionne un motif incluant le bouchon de la bouteille et un autre incluant la majeure partie de son étiquette.

Une première version de cet algorithme se focalise sur la recherche de motifs sous forme de disques, afin de ne pas devoir s'occuper de leur orientation dans le processus de suivi. Pour chaque diamètre de disque possible, depuis une taille minimale jusqu'au diamètre du plus grand disque inscrit dans l'objet de référence, l'algorithme considère chaque position d'un tel disque comme un motif potentiel et, pour chacun, il calcule une valeur déterminée par la pondération des critères que nous avons cités plus haut. Un autre avantage des disques est quelle que soit leur orientation, le nombre de pixels de la fenêtre de recherche n'appartenant pas à l'objet de référence est constant (pour une taille donnée). Une autre version de cet algorithme se focalise, quant à elle, sur la recherche de motif carrés car les méthodes utilisées sont optimales pour les cibles rectangulaires, et travailler avec des cibles carrées confère, à l'algorithme du CHS, une plus grande robustesse à l'orientation.

Ce processus d'identification des motifs a cependant quelques lacunes. D'une part, il nécessite environ 5 minutes pour traiter un objet d'une taille de 50 pixels sur 100. D'autre part, limiter les motifs candidats à des disques, ou à des carrés, risque de ne pas mettre en évidence les motifs les plus pertinents pour l'objet analysé. Certaines zones caractéristiques de la cible peuvent avoir une forme quelconque. Un prétraitement de type segmentation couleur de l'image permettrait de créer automatiquement un catalogue de motifs à évaluer. Nous nous sommes contentés d'envisager des motifs carrés et circulaires qui sont suffisants pour valider la méthode de suivi de motifs (section 6.4.3).

En prenant l'exemple d'une bouteille, que nous réutiliserons comme objet de référence par la suite, la figure 6.10 nous montre 2 motifs sélectionnés par l'algorithme.

FIG. 6.10 – Un objet de référence et deux motifs sélectionnés par l'algorithme

Une fois ces motifs sélectionnés, nous pouvons maintenant envisager de suivre l'objet au travers de ces motifs.

6.4.3 L'algorithme de suivi de motifs

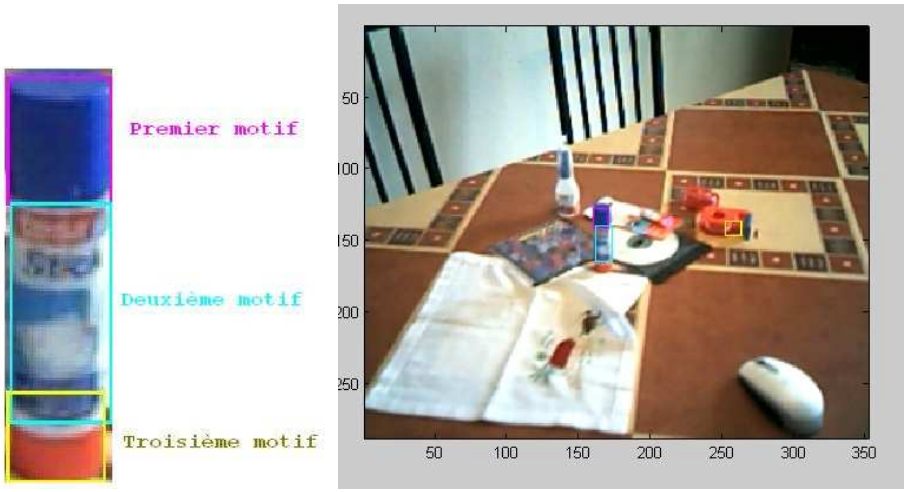
Pour suivre un motif particulier, nous allons utiliser l'algorithme du CHS en utilisant les histogrammes propres (section 6.2.3) [1], mais en se limitant à la partie concernant la localisation. A nouveau, nous allons nous servir de l'estimation de la taille et de l'orientation de l'objet dans la scène précédente pour localiser les motifs que nous avons sélectionnés.

Il est important de rechercher l'objet cible et d'ensuite procéder à la rechreche des sous-objets à l'intérieur de celui-ci. Si nous nous contentions de rechercher les motifs dans la scène il nous faudrait prendre comme hypothèse qu'aucune autre zone de la scène n'aie une distribution identique à l'un des motifs recherchés. La figure 6.4.3 illustre ce phénomène. Un sous objet es mal localisé, car dans la scène il existe un objet ressemblant plus au motif que le motif cible.
En recherchant l'objet cible en fonction de la taille et l'orientation estimée à l'itération précédente, et en restreignant la recherhce de motifs au voisinage de l'objet cible, cette hypothèse est elle aussi restreinte au voisinage de l'objet. Nous utilisons donc l'objet de référece pour localiser l'objet cible, et les motifs pour en affiner l'orientation.

Par ailleurs, nous allons suivre des motifs et plus seulement l'objet de référence dans son ensemble. Par conséquent, quand la caméra sera loin de l'objet de référence, les motifs seront de très petite taille dans la scène. La localisation d'objets de petite taille s'avère difficile, car les histogrammes contiennent beaucoup moins de pixels. Par conséquent, du bruit dans l'image, suite à un masquage partiel de l'objet recherché, ou pour d'autres raisons, peut grandement affecter la similarité de l'objet cible.

Afin d'éviter qu'un motif mal localisé entraine une mauvaise estimation de la taille et de l'orientation, nous avons mis en place une série de techniques pour gérer ces erreurs, notamment une tehcnique permettant de détecter quel motif à mal été localisé afin de ne pas en tenir compte dans l'estimation de taille et d'orientation.

[1] La version du CHS optimisé n'était pas achevée (contrainte de taille minimale de l'objet recherché),nous avons mis au point une variante qui lève cette contrainte.



(a) Baton de référence et motifs choisis(manuellement)

(b) Localisation erronée d'un des motifs

FIG. 6.11 – Un des motifs est mal localisé car une zone de l'image ressemble plus au motif de référence

Nous avons testé cet algorithme de suivi à la fois avec des motifs sélectionnés manuellement et des motifs sélectionnés par l'algorithme que nous avons décrit précédemment. Les résultats obtenus sont similaires dans les deux cas et plutôt encourageants. Cependant, lorsque la cible est trop lointaine la taille des motifs ne permet plus une détection fiable, cette méthode n'est donc plus envisageable.

6.4.4 L'estimation de la taille et de l'orientation

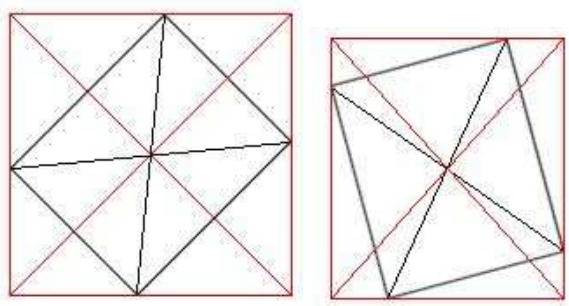
Supposons que nous ayons identifié les différents motifs de notre objet de référence dans la scène. Comment, à partir de ces motifs, pourrons-nous déterminer la taille et l'orientation de l'objet cible ?

En ce qui concerne l'estimation de la taille, l'approche la plus simple consiste à comparer les distances entre les motifs dans la scène avec les distances entre motifs dans l'image de référence. Le rapport entre ces distances correspond au facteur d'échelle entre l'objet cible et l'objet de référence.

Si en pratique nous allons plutôt utiliser un système de calcul matriciel nous permettant de déterminer à la fois la position de l'objet, sa taille et son orientation, cette approche permet de mettre en évidence deux points importants.

- D'une part, nous ne pouvons nous baser sur les distances entre les positions des fenêtres de recherche associées aux motifs, qui varient en fonction de l'orientation. Si nous utilisions des masques de l'objet cible pour la localisation, nous pourrions utiliser les distances entre les centres de gravité des motifs. Toutefois, comme nous l'avons dit dans la section 6.2, nous avons opté pour une méthode de localisation n'utilisant pas de masques afin d'éviter des suppléments de calcul. En nous restreignant aux motifs de forme rectangulaire et circulaire, nous

pouvons cependant nous baser sur les distances entre les centres des motifs, qui correspondent aux centre des fenêtre de recherche. Cette propriété est évidente pour les motifs circulaires, et est démontrable pour les fenêtres de recherches carrées. Nous nous contenterons ici d'illustrer ce phénomène sur la figure.



(a) (b)

FIG. 6.12 – Le centre des fenêtres de recherches correspond au centre des objets dans le cas des objets rectangulaires.

- D'autre part, plus nos motifs seront décentrés les uns des autres, plus notre suivi sera robuste. En prenant le cas de deux motifs dont les centres sont éloignés d'un pixel seulement, si la localisation de l'un d'eux est décalée d'un pixel suite à un changement de la taille ou de l'orientation par rapport à l'image précédente, il en résultera une estimation de la taille totalement erronée. Si, au contraire, la distance entre les centres de nos motifs est grande, une faible erreur de localisation, due au changement de taille ou d'orientation, aura un impact quasiment nul sur la nouvelle estimation. Ce phénomène met en évidence un facteur supplémentaire qui devrait être pris en compte lors du choix des motifs : la distance entre les centres des motifs sélectionnés.

D'une manière générale, nous estimerons la taille et l'orientation de la cible par rapport à la caméra par un simple calcul matriciel. En ne prenant en compte que les rotations dans le plan, les translations et les changements d'échelle, la correspondance entre les motifs dans une scène et ces mêmes motifs dans l'objet de référence peut être représentée par l'équation suivante :

$$\begin{bmatrix} u & v \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} * \begin{bmatrix} s*\cos(\theta) & -s*\sin(\theta) \\ s*\sin(\theta) & s*\cos(\theta) \\ tx & ty \end{bmatrix}$$

Où

- s est le facteur d'échelle,
- θ est l'angle de la rotation,
- tx et ty sont respectivement les translations selon x et y.

$$\begin{cases} u = \begin{bmatrix} x & y & 1 & 0 \end{bmatrix} * \begin{bmatrix} s * \cos(\theta) \\ s * \sin(\theta) \\ tx \\ ty \end{bmatrix} \\ \\ v = \begin{bmatrix} y & -x & 0 & 1 \end{bmatrix} * \begin{bmatrix} s * \cos(\theta) \\ s * \sin(\theta) \\ tx \\ ty \end{bmatrix} \end{cases}$$

Nous voulons retrouver l'angle de rotation, les translations et le rapport d'échelle entre l'objet cible dans la scène et l'objet de référence. Nous avons 4 inconnues, ce qui implique qu'avec un minimum de 2 correspondances, donc un minimum de deux motifs suivis, nous pouvons combiner les équations u et v pour former un système linéaire dont la résolution nous fournira la position de l'objet cible dans l'image, son orientation et sa taille. Pour n motifs, nous avons le système suivant :

$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \\ v_1 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 \\ y_1 & -x_1 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ y_n & -x_n & 0 & 1 \end{bmatrix} * \begin{bmatrix} s * \cos(\theta) \\ s * \sin(\theta) \\ tx \\ ty \end{bmatrix}$$

De manière plus simple, ce système peut se réécrire par

$$U = X * r$$

Il nous suffit maintenant de résoudre ce système d'équation pour déterminer la taille et l'orientation de l'objet cible. Si nous disposons de deux motifs, la résolution de ce système est simple. Si, par contre, nous avons davantage de motifs, la résolution de ce système nous fournira la solution minimisant l'erreur via la méthode des moindres carrés, c'est-à-dire la solution minimisant la norme du vecteur $[U - X * r]$.

6.4.5 Conclusion

Dans cette section, nous avons envisagé, pour estimer la taille et l'orientation de l'objet, une méthode originale qui a l'avantage de permettre de déterminer assez précisément la taille et l'orientation de l'objet cible, même en cas de localisation imprécise des motifs.
Les tests que nous avons effectués pour cet algorithme de suivi ont montré des résultats plutôt encourageants. L'utilisation des motifs nous permet d'estimer assez précisément la taille et l'orientation de l'objet cible pour assurer un suivi robuste, si toutefois nous disposons de motifs dont les centres sont distants les uns des autres. Cependant, lorsque l'objet cible est éloigné, la localisation précise des motifs s'avère difficile. Il faudra peut-être envisager une autre technique pour estimer la position des objets éloignés, ou tout simplement ne pas se préoccuper de leur orientation tant

que la caméra est éloignée de l'objet cible.

Pour le moment, notre algorithme ne manipule que des motifs rectangulaires ou des disques. Cependant, au moyen de la technique utilisant des masques, nous pourrions localiser des motifs de forme quelconque et utiliser le centre de gravité des motifs pour déterminer la taille et l'orientation de l'objet cible.

6.5 Conclusion

Dans ce chapitre, nous avons analyser le comportement de l'algorithme du CHS lorsque la forme des objets est quelconque. Lorsque les formes ou les variations d'inclinaison ne permettent plus d'assurer un suivi robuste des objets, il est possible d'introduire des techniques complémentaire. Nous avons élaboré plusieurs algorithmes pour suivre les objets quelconques et opté pour une méthode qui nous permet un traitement rapide des images.

Nous avons ensuite essayé d'estimer l'orientation de l'objet cible par rapport à la caméra. N'ayant aucune trouvé aucune technique efficace et rapide pour estimer l'orientation, nous nous sommes tournés vers la sélection de motifs au sein de l'objet de référence, motifs que nous allons suivre pour pouvoir évaluer la taille et l'orientation de l'objet cible. Cette méthode s'avère prometteuse, mais présente des limites quant à la taille des motifs utilisés.

Chapitre 7

Validation sur prototype robotique

7.1 Objectifs

Nous avons consacré la dernière semaine de notre stage à la réalisation d'expériences de suivi temps réel sur des mini-prototypes robotiques, et ce dans un double objectif.

D'une part, celles-ci nous ont permis de valider le bon fonctionnement de nos algorithmes (indexation, intersection d'histogrammes, estimation de la taille, ...) dans des conditions réelles. D'autre part, elles nous ont permis de démontrer que les performances des différents algorithmes et des implémentations que nous en avons réalisées étaient suffisantes pour être utilisées dans un contexte d'asservissement temps réel.

Nous présentons ici brièvement la configuration expérimentale que nous avons mise au point et commentons les résultats qui en sont ressortis.

7.2 Configuration expérimentale

L'expérience consistait en le suivi d'un premier robot (R_1), portant une cible et effectuant une trajectoire prédéfinie, par un second (R_2) équipé d'une caméra numérique reliée à un ordinateur et pilotés par celui-ci via une liaison infrarouge. La figure 7.1 schématise cette configuration.

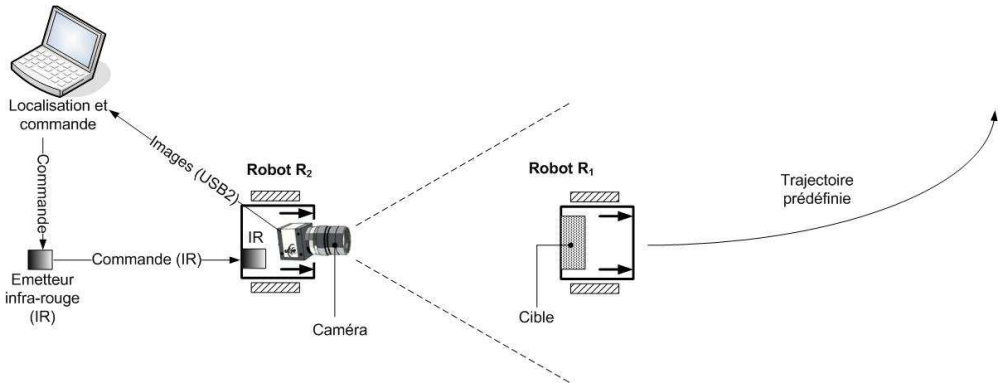


FIG. 7.1 – Configuration expérimentale

7.2.1 Traitement d'image et commande

Toute la partie logicielle relative à l'acquisition, au traitement d'image et à la commande du robot R_2 est supportée par un script Matlab (annexe A), exécuté sur notre machine de test.

- L'acquisition des images sur la caméra de R_2 est réalisée via une API Matlab que nous avons définie et implémentée (7.2.3). Celle-ci permet de configurer la caméra et d'acquérir des images RGB 24bpp à cadence vidéo dans l'environnement Matlab. Le temps d'acquistion d'une image en 640 × 480 pixels est d'environ 12ms.
- L'indexation est réalisée par l'algorithme que nous avons développé en 3.5.3. Son temps d'exécution dépend linéairement de la taille de l'image ; une image en 640 × 480 pixels est indexée moins d'une milliseconde.
- Nous avons également apporté certaines corrections à l'implémentation calculant la surface de similarité, particulièrement pour la localisation de cibles de petite taille. La recherche d'une cible de 80 × 50 pixels dans une image de 640 × 480 pixels prend environ 19ms.
- L'algorithme d'estimation de la taille, étant toujours en développement, n'a encore été implémenté qu'en script Matlab. Son temps d'exécution dépend linéairement de la taille de la fenêtre et de la variation de taille de la cible. Pour une cible de 65 × 55 pixels (dans le pire des cas) son temps d'exécution est de 48ms. Une implémentation en C permettrait de réduire considérablement ce temps déjà raisonnable.

7.2.2 Prototypes robotiques

Nous disposions de deux prototypes robotiques Lego® possédant 2 roues motrices et une roue arrière flottante. Les deux roues motrices sont actionnées indépendamment l'une de l'autre par deux moteurs asservis par des capteurs de position, permettent aux robots d'avancer et de tourner, suivant la vitesse des moteurs. Chaque robot était équipé d'un microcontrôleur programmable capable d'exécuter plusieurs tâches simultanées nous permettant de commander les robots. Une liaison infrarouge intégrée à chaque contrôleur nous permet d'une part de les programmer et d'autre part de communiquer avec eux, depuis l'ordinateur.

Robot R_1

Le microcontrôleur du premier robot exécute "aveuglément" un programme lui faisant décrire une trajectoire prédéfinie. Une tâche de synchronisation contrôlant l'évolution de la valeur des capteurs permet au robot d'effectuer des mouvements précis, par exemple décrire une droite parfaite. Enfin, celui-ci est muni d'une cible permettant à l'autre robot de le localiser assez facilement même lorsque les angles de vues changent radicalement.

Nous avons programmé trois types de trajectoires. Une ligne droite, permettant essentiellement de valider le maintien de distance entre les robots. Une ligne brisée, permettant de valider le recentrage et enfin un "huit", dans un environnement plus riche permettant de valider les deux aspects combinés ainsi que la tolérance aux variations de luminosité.

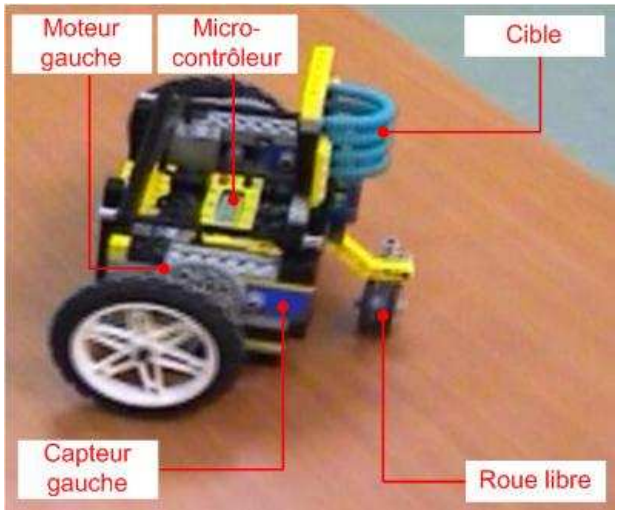
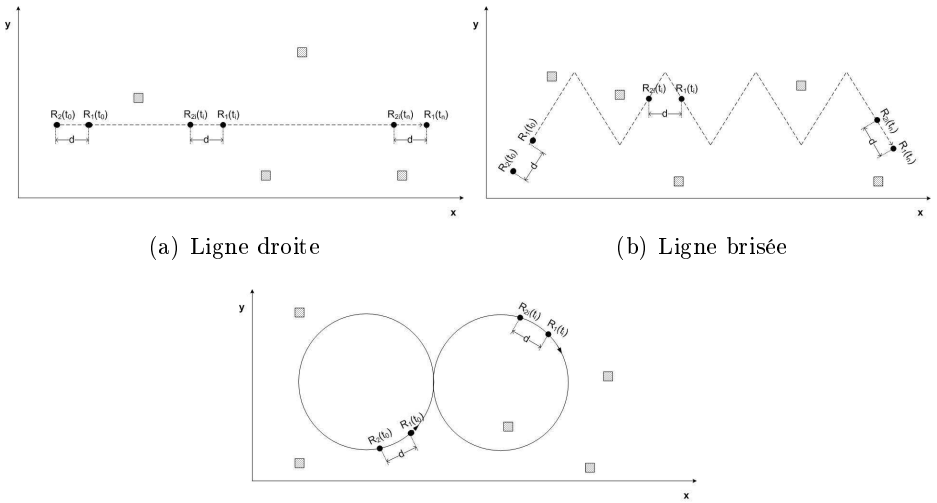


FIG. 7.2 – Robot R_1 qui effectue une trajectoire prédéfinie

Robot R_2

Le microcontrôleur du second robot exécute un programme intégrant les commandes reçues de l'ordinateur qui effectue la localisation et les corrections à apporter en conséquence. La stratégie de suivi que nous avons mise en oeuvre est relativement élémentaire. Le robot R_2 avance de façon à se tenir à distance constante du robot R_1, en exploitant la valeur du facteur d'échelle calculée par l'algorithme d'estimation de taille. Lorsque la cible placée sur R_1 dévie trop fortement du centre de l'image vue par la caméra de R_2, une correction sur la trajectoire de celui-ci est appliquée pour recentrer la cible dans l'image, pour ensuite continuer à avancer.



(a) Ligne droite

(b) Ligne brisée

(c) Huit

FIG. 7.3 – Différents types de trajectoires

7.2.3 Caméra

Description

Durant notre stage, et particulièrement lors des expériences avec les robots, nous avons travaillé avec une caméra industrielle IDS uEye 1210-C, possédant un capteur CMOS (derrière un filtre de Bayer) d'une résolution de 640 par 480 pixels et reliée à l'ordinateur via une liaison USB2.

Son avantage par rapport à une webcam classique réside dans le contrôle que l'on a de la caméra. Les drivers des webcam classiques ne permettent généralement pas de désactiver toutes les corrections automatiques, effectuées par le matériel ou par le driver lui-même (balance des blancs, luminosité, ...). Si celles-ci permettent de rendre les images agréables à l'oeil humain (ce qui est appréciable dans le cadre d'une application de visio-conférence), elles se révèlent toutefois gênantes en ce qui nous concerne. Nous ne voulons en effet pas que les images soient corrigées mais plutôt qu'elles rendent compte de la scène telle qu'elle est.

API Matlab

A notre arrivée, aucune API efficace n'existait pour utiliser cette caméra sous Matlab. Nous en avons donc défini une API et implémenté ses différentes fonctionnalités dans une librairie C à l'aide du SDK[1] fourni par IDS.

Celle-ci permet d'une part de contrôler (initialisation, comptage d'images, ...) et configurer (format et taille d'image, fenêtrage, cadence d'acquisition, luminosité, contraste, gamma, gain, ...) la caméra. D'autre part, elle permet d'acquérir des images dans l'environnement Matlab (au format image Matlab, ie : matrice tridimensionnelle, un plan par couleur) à cadence vidéo.

[1] Software Development Kit

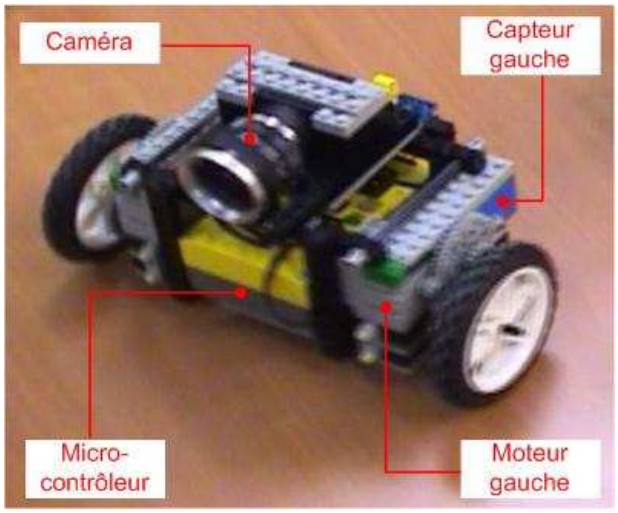


FIG. 7.4 – Robot R_2 qui effectue le suivi

Stratégie d'acquisition Nous avons opté pour une acquisition travaillant avec un anneau de A buffers (zones mémoires) dans lesquels vont successivement prendre place les acquisitions des différentes images. La première acquisition se fait dans le premier buffer de l'anneau, la deuxième dans le suivant et ainsi de suite jusqu'au dernier. Une fois le dernier buffer atteint, le premier est réutilisé (son contenu est donc écrasé) pour acquérir l'image suivante.

Cette stratégie permet de constamment garder A-1 images prêtes à être traitées, et donc de ne pas subir le délai imposé par la numérisation, le transfert et le pre-processing par le driver. Nous avons choisi de travailler avec une valeur de A=2 (ie : en double buffering), l'acquisition et le traitement prenant alternativement place dans deux buffers distincts (figure 7.5). Des valeurs de A>2 permettent de prendre un retard temporaire dans le traitement par rapport à ce qui est acquis. Cependant, dans notre cas, si le traitement ne s'effectue pas suffisamment vite, il est préférable de laisser tomber des images plutôt que d'accumuler un retard qu'il ne sera pas possible de rattraper.

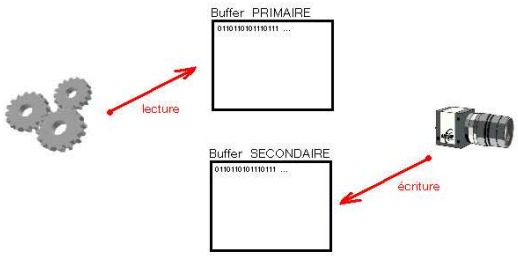


FIG. 7.5 – Acquisition en double buffering

Pour indication, le temps moyen d'acquisition d'une image 640 × 480 en RGB 24bpp est de 12ms, ce qui laisse 28ms pour effectuer le reste des traitements, dans les cas où l'on veut tenir une cadence de 25 images/secondes.

Si les images sont traitées à une cadence supérieure à celle avec laquelle la caméra délivre les images, une même image peut être traitée plusieurs fois, ce qui n'est évidemment pas souhaitable. Ainsi, afin d'éviter des traitements inutiles, nous avons synchronisé le transfert d'image vers l'environnement Matlab sur l'événement (Windows) "nouvelle image" générée par la caméra à chaque fois qu'une numérisation vers un buffer vient d'être terminée. Ceci permet en outre de laisser disponibles les ressources de la machine pour l'exécution d'autres tâches, comme celle de commande par exemple.

Dispositif optique

Mise au point Dans les applications envisagées, la distance entre la caméra et l'objet à localiser sera amenée à varier. Etant donné que la caméra ne fait aucune mise au point (son objectif n'étant pas motorisé), l'image de l'objet suivi deviendra floue lorsque cette distance variera d'une valeur supérieure à environ la moitié de la profondeur de champ ([2], page 98) par rapport à sa position initiale, où la première mise au point a été effectuée.

Aussi, même si cette opération de mise au point était possible, il faudrait encore savoir sur quelle région effectuer la focalisation ; sur celle de l'objet certes, mais jusque là encore inconnue puisque c'est exactement ce que l'on recherche ! La solution consisterait alors à travailler avec un objectif de distance hyperfocale (et donc de distance focale) la plus courte possible. Lorsque le point est fait sur un objet situé à la distance hyperfocale, la profondeur de champ s'étend de la moitié de cette distance jusqu'à l'infini. En ne travaillant qu'avec des objets situés à une distance supérieure à cette demi-distance hyperfocale et avec un objectif initialement réglé pour être au point sur des objets se trouvant à cette distance hyperfocale, on pourrait sans problème omettre ce processus de focalisation. La distance hyperfocale peut être calculée à l'aide de la formule suivante :

$$d_{hf} = \frac{0,001.f^2}{N.c} \tag{7.1}$$

où
- d_{hf} = distance hyperfocale (m)
- f = distance focale (mm, précisé sur objectif)
- N = nombre d'ouverture du diaphragme (sans dimension, gravé sur la bague du diaphragme)
- c = diamètre du cercle de confusion (mm, dépend du capteur : annexe B)

Le tableau ci-dessous indique les valeurs moyennes de référence des diamètres de confusion pour différents formats de capteurs.

Plus de précisions sur toutes ces notions peuvent être trouvées dans [2], pages 69-101], et plus particulièrement sur la mise au point (tirage optique, page 90) et la distance hyperfocale (page 100).

Ouverture du diaphragme Le fait que le diaphragme ne soit lui non plus pas motorisé ne pose pas un réel problème. En effet, d'après [2], page 94 l'équivalent d'une division de diaphragme (une graduation, division par racine de 2) s'obtient par une augmentation du gain de 6dB, ce qu'il est possible de réaliser matériellement sur la caméra.

7.3 Résultats

Notre implémentation est parvenue à assurer le suivi et la commande de R_2 sur les trois types de mouvements que nous avons envisagés, avec une cadence de traitement d'environ 15 images/seconde.

Le maintien d'une distance constante entre les deux robots nous a permis de confirmer le bon fonctionnement de notre algorithme d'estimation de taille. Des traces vidéo de ces expériences peuvent êtres consultées sur le support numérique annexé au mémoire.

Conclusion et perspectives

Tout au long de ce mémoire, nous nous sommes intéressés au problème de la localisation d'une cible au sein d'images couleurs. Nos travaux se sont particulièrement focalisés sur l'élargissement du potentiel de l'algorithme du *CHS*, dont la version de base présentait certaines limites.

Afin de bien délimiter le problème, nous avons commencé par poser le cadre d'objectifs, d'hypothèses et de contraintes dans lequel le laboratoire MIPS veut inscrire ses développements en matière de vision. Nous avons ensuite situé notre travail par rapport à celui du laboratoire et à celui d'André et Frippiat pour définir un point de départ de nos investigations.

Dans un premier temps, nous avons voulu consolider le choix de la méthode du *CHS* par rapport aux autres approches pour la localisation d'objets dans un contexte d'asservissement visuel en environnement non contrôlé.

A cette fin, nous avons dressé un état de l'art que nous avons confronté avec le cadre de développement posé. Cette analyse préalable nous a d'une part permis de rentrer dans le domaine du traitement d'images, jusqu'alors inconnu pour nous, et d'autre part elle nous a servi de source d'inspiration des développements que nous avons menés.

Nous avons ensuite présenté plus largement l'algorithme de base du *CHS*. Nous avons discuté des principales optimisations qui le rendaient applicable en temps réel (différence d'histogrammes, SIMD) et en avons proposé d'autres supplémentaires (multi-threading, fenêtrage), tirant parti de possibilités matérielles. Nous l'avons ensuite analysé dans l'objectif de mettre en évidence ses problèmes et ses limites. De cette évaluation sont ressortis un certain nombre d'aspects à travailler tels que

- le choix de l'espace couleur et sa quantification,
- l'optimisation de la signature couleur comme modèle de suivi,
- la sensibilité de l'intersection d'histogrammes aux conditions d'illumination,
- l'estimation de la taille de la cible en vue d'un ajustement dynamique,
- la généralisation de l'algorithme au suivi de cibles de formes quelconques.

Nous avons abordé ces différents points, tout d'abord en explorant la littérature et les perspectives proposées par nos prédécesseurs, puis en menant nos propres réflexions qui, dans certains cas ont amené des solutions satisfaisantes (mais pas toujours complètes).

Nous avons commencé par généraliser la notion de quantification afin de maîtriser plus finement la construction des classes sur lesquelles repose l'histogramme couleur, pièce maîtresse du *CHS*. Ce travail préliminaire nous a permis d'aborder plus précisément les autres aspects à travailler tels que l'optimisation de la signature couleur ou la tolérance aux variations de luminosité. Afin de pouvoir travailler en temps réel avec ces quantifications, nous avons également développé et implémenté des algorithmes d'indexation rapide.

Nous avons, par ailleurs, abordé le problème de la construction et de l'optimisation de la signature couleur (niveau de détail, restriction de la quantification à l'ensemble des points couleurs de la référence, utilisation d'une classe d'exclusion,...). Nous avons également commencé à développer une technique permettant d'optimiser la représentation d'une image couleur par un histogramme. Si celle-ci ne fournit pas toujours de bons résultats, nous encourageons vivement la suite de son développement, son utilité étant indéniable.

De nombreux travaux ont déjà mis en évidence la sensibilité des histogrammes aux variations des conditions d'éclairage. Nous avons orienté l'étude de cette première problématique suivant deux grands axes différents.

Dans un premier temps, nous avons étudié la *tolérance* que pouvait apporter le choix de l'espace et sa quantification. Nous avons vu que choisir un espace en vue de le quantifier dans une direction privilégiée n'apportait finalement pas une solution intéressante dans la mesure où elle ne contrecarrait qu'un type particulier de variation. Le choix de la quantification était cependant très important puisque c'est elle qui fixe la tolérance de l'algorithme.

Dans un second temps, nous avons essayé d'étendre cette tolérance à la prise en charge de plus grandes variations, pour amener une certaine *indépendance*. Nous avons préalablement étudié plusieurs modèles de variations de l'illumination pour ensuite aborder la construction de transformations de normalisation (Finlayson) permettant d'annuler les effets des variations de luminosité dans les images. Dans une optique d'intégration au *CHS*, nous avons procédé à différentes vérifications. Nous nous sommes assurés du caractère indépendant des images normalisées et avons vérifié que celles-ci permettaient de construire des histogrammes aussi discriminants que ceux reposant sur des images brutes. Enfin, des tests de performances nous ont permis de montrer que l'application de ces transformations était envisageable en temps réel.

Nous avons proposé une extension du *CHS* intégrant la normalisation. Cependant, la nécessité de normaliser chaque fenêtre de recherche empêche d'utiliser les optimisations par histogramme différentiel, rendant le *CHS* applicable en temps réel. Des recherches sont donc encore à effectuer dans ce sens.

La deuxième grande problématique que nous avons abordée concernait la gestion dynamique des variations de taille apparente de la cible. Nous avons commencé par discuter d'une extension du CHS permettant de prendre en compte les

re-dimensionnements. Nous avons vu que celle-ci reposait sur une hypothèse itérative permettant d'utiliser le facteur d'échelle calculé à une itération pour effectuer une première localisation à l'itération suivante. Cette dernière est ensuite corrigée avec l'estimation fournie par un algorithme calculant le facteur d'échelle en analysant une fenêtre candidate.

Comme l'avaient suggéré nos prédécesseurs, nous avons approfondi l'étude de la méthode du *CHF* réalisant cette analyse. Si cette méthode donne de bons résultats dans la plupart des cas, l'impossibilité de déterminer ses paramètres (cfr. étude de cas théorique appuyée par des images réelles et de synthèse) nous a incités à proposer une nouvelle technique dans laquelle l'estimation de taille se fait par l'étude des propriétés d'une famille d'indicateurs. Néanmoins, celle-ci repose encore sur une hypothèse de travail qu'il conviendrait de lever dans le futur, les résultats étant encourageants. Nous avons démontré son potentiel dans les expériences de suivi sur prototype robotique réalisées en fin de stage.

La troisième et dernière grande problématique que nous avons abordée est celle de la généralisation de la méthode du *CHS* au suivi de cibles de formes quelconques. Une technique utilisant des masques et une autre utilisant des histogrammes propres se sont révélées efficaces pour suivre des objets quelconques, chacune ayant ses avantages et ses inconvénients. La technique que nous avons retenue, à savoir celle des histogrammes propres, a ensuite été utilisée dans un procédé visant à estimer la taille et l'orientation de la cible au moyen de sous-objets (ie : des motifs particuliers) que cette dernière présente. Ce procédé, bien qu'il ne soit encore qu'à ses débuts, se révèle assez prometteur. Il fournit, dans la plupart des cas, une estimation précise de la taille et de l'orientation. Cependant, certains points lui font encore défaut, notamment la difficulté à suivre des motifs de trop petite taille et l'absence d'une méthode efficace de sélection automatique des motifs. Au vu des résultats obtenus, nous estimons qu'il serait intéressant de poursuivre son développement, d'autant plus que ce procédé permettrait, moyennant un critère apte à déterminer si un motif a été mal localisé, de détecter si la cible est partiellement masquée ou partiellement absente du champ de la caméra.

Afin de démontrer l'efficacité et les possibilités des algorithmes développés, nous avons consacré la dernière semaine de notre stage à la mise en oeuvre d'une expérience de suivi sur une mini plate-forme robotique, dont les résultats plutôt concluant peuvent être visionnés sur les séquences vidéos se trouvant sur le support numérique annexé à ce document.

L'étude des différents problèmes que nous avons abordés jusqu'ici est encore loin d'être terminée. Les solutions que nous avons apportées n'en sont encore qu'à leurs débuts et les pistes de réflexion que nous avons ouvertes mériteraient d'êtres explorées plus profondément. Par ailleurs, notre travail a soulevé de nombreuses questions, pas moins intéressantes, mais malheureusement restées en suspens. Nous conclurons en les évoquant brièvement pour que celles-ci trouvent peut être un jour une réponse.

Nous avons vu que l'algorithme du CHS trouvait toujours une solution, mais

pas nécessairement la bonne (cible non présente dans la scène, forte variation de luminosité, ...). Il conviendrait de prolonger les travaux d'André et Frippiat relatifs à la recherche d'indicateurs permettant de séparer les bonnes solutions des mauvaises.

Une autre perspective repose sur le constat que le mouvement d'une cible à travers une séquence d'images n'est (généralement) pas aléatoire. Il pourrait être intéressant d'extrapoler son mouvement pour effectuer une analyse plus locale qui permettrait d'une part d'accroître la cadence de traitement et d'autre part de rendre la solution plus fiable.

Enfin, la dernière question, qui n'est pas des moindres, concerne les changements de perspectives. Bien que l'algorithme du *CHS* y soit relativement tolérant, de plus importantes variations peuvent induire des déformations considérables dans l'image de la cible et/ou y amener des parties de la surface qui n'étaient pas visibles lors de sa capture. La question de la gestion des déformations et de la mise au point d'un modèle plus général capable de décrire l'intégralité de la surface de la cible pourrait, à notre sens, faire l'objet d'un futur travail tout aussi passionnant.

Bibliographie

[1] Arnaud André and Sébastien Frippiat. Traitement d'images couleur : Localisation d'objets en temps réel. Master's thesis, Facultés Universitaires Notre-Dame de la Paix, 2005.

[2] Philippe Bellaiche. *Les secrets de l'image vidéo*. Eyrolles, Paris, France, 5 edition, 2004.

[3] Lucien Birgé and Yves Rozenholc. How many bins should be put in a regular histogram. *ESAIM : Probability and Statistics*, 2006.

[4] Andrew Blake and Michael Isard. *Active Contours : The Application of Techniques from Graphics, Vision, Control Theory and Statistics to Visual Tracking of Shapes in Motion*. Springer-Verlag, New York, USA, 1 edition, Août 2000.

[5] Gary R. Bradski. Computer Vision Face Tracking For Use in a Perceptual User Interface. *Intel Technology Journal*, (Q2) :15, 1998.

[6] M.H. Brill. A device performing illuminant-invariant assessment of chromatic relations. *Journal of Theoretical Biology*, page 473, 1978.

[7] Jean-Luc Buessler, Jean-Philippe Urban, Gilles Hermann, and Hubert Kihl. Color Histogram Similarity for Robot-Arm Guiding. *International Conference on Complex Systems, Intelligence and Modern Technology Applications (CSIMTA 2004)*, pages 515–519, 2004.

[8] Jean-Luc Buessler, Jean-Philippe Urban, Gilles Hermann, and Hubert Kihl. Color Histogram Algorithms for Visual Robot Control. *International Journal of Robotics and Automation (Special Issue on Colour Image Processing and Analysis for Machine Vision)*, pages 86–93, 2005.

[9] Jean-Luc Buessler, Jean-Philippe Urban, and Hubert Kihl. Color Histogram Footprint technique for visual object tracking. *IEEE Conference on Control Applications (CCA 2005)*, pages 761–766, 2005.

[10] L.D. Cohen. On active contour models and balloons. *Computer Vision Graphics, and Image Processing : Image Understanding*, Vol. 53(No. 2) :pp. 211Ű218, 1991.

[11] Rogerio Feris, Volker Krüger, and R. Cesar Jr. Efficient Real-Time Face Tracking in Wavelet Subspace. *Proceedings of the Int. Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-Time Systems*, 2001.

[12] Graham Finlayson. Colour Invariant, http ://www.sys.uea.ac.uk/Research /groups/compvis/ColourInvariants/ColourInvariants.htm.

[13] Graham Finlayson. *Coefficient Colour Constancy*. PhD thesis, Simon Fraser University, 1995.

[14] G. Florou and R. Mohr. What accuracy for 3D measurements with cameras ? *Proceedings of the 13th International Conference on Pattern Recognition*, I :354358, 1996.

[15] Brian V. Funt and Graham D. Finlayson. Color constant - Color indexing. *IEEE Transactions on pattern analysis and machine intelligence*, 17(5), May 1995.

[16] B. Schiele G.D. Finlayson and J. Crowley. Comprehensive colour image normalization. *eccv98*, pages 475–490, 1998.

[17] Mark S. Drew Graham D. Finlayson and Brian V. Funt. Spectral Sharpening : sensor transformations for improved color constancy. *Journal of the Optical Society of America*, A(11(5)) :1553–1563, 1994.

[18] P. Gros, G. Mclean, R. Delon, R. Mohr, C. Schmid, and G. Mistler. *Utilisation de la couleur pour l'appariement et l'indexation d'images.* PhD thesis, Institut National de Recherche en Informatique et en Automatique (INRIA), Septembre 1997.

[19] J. Huang, S. R. Kumar, M. Mitra, and Wei-Jing Zhu. Spatial color indexing and applications. In *IEEE International Conference on Computer Vision ICCV*, pages 4–7, Jan 1998.

[20] A. Jacquot, Sturm P., and Ruch O. Adaptative Tracking of Non-Rigid Objects Based on Color Histograms and Automatic Parameter Selection. *IEEE Workshop on Motion and Video Computing*, pages pp 103–109, January 2005.

[21] Larousse. *Petit Larousse illustré.* Editions Larousse, 1994.

[22] S. Lefevre, B C. Fluck, Maillard, and N. Vincent. Un modèle de contour actif pour le suivi rapide d'objets en mouvement. Application au suivi de joueurs de football. *18e Colloque GRETSI sur le traitement du Signal et des Images*, pages 10–13, septembre 2001.

[23] J. P Lewis. Fast Template Matching. *Vision Interface*, pages 120–123, 1995.

[24] Ludovic Macaire. Exploitation de la couleur pour la segmentation et l'analyse d'images. Master's thesis, Université des Sciences et Technologies, 2004.

[25] J. B. T. M. Roerdink and Arnold Meijster. The Watershed Transform : Definitions, Algorithms and Parallelization Strategies. *Fundam. Inform.*, 41(1-2) :187–228, 2000.

[26] Jean Serra and Jesus Angulo. Traitement des images de couleur en représentation luminance/saturation/teinte par norme L1. *Traitement du Signal*, avril 2004.

[27] M. Stricker, M. Swain. The capacity of color histogram indexing. *Computer Vision and Pattern Recognition*, pages 704–708, Jun 1994.

[28] M.A. Stricker and M. Orengo. Similarity of color images. *Conference on Storage and Retrieval for Image and Video Databases V*, SPIE(2420) :381, 1995.

[29] M. Swain and M. Ballard. Color indexing. *International Journal of Computer Vision (IJCV)*, 7(1) :11– 32, 1991.

[30] Alain Trémeau, Christine Fernandez-Maloigne, and Pierre Bonton. *Image numérique couleur : De l'acquisistion au traitement.* Dunod, Paris, France, 2004.

[31] Haim J. Wolfson and Isidore Rigoutsos. Geometric Hashing : An Overview. *IEEE Computational Science & Engineering*, pages 10–21, Octobre - Décembre 1997.

[32] Xavier Zimmermann. Traitement d'images couleur pour le suivi de cible : Application au positionnement d'un bras robotique. Rapport de DEA. Rapport de dea, Ecole Supérieure des Sciences Appliquées pour Ingénieur de Mulhouse, Mulhouse, 2004.

Annexe A

Script de localisation/commande

```
0001 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
0002 %% Pilotage Légo : suivi d'un mobile
0003 %% Version finale
0004 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
0005 close all ; clc ; clear scRec
0006
0007 % DLL utilisées
0008 % chist14 : Calcul de la surface de similarité v1.4
0009 % uEyeMat : API caméra uEye
0010 % img2ind64 : Algorithme d'indexation rapide
0011 % rcx2b : Interface de commande robot
0012
0013 %% PARAMETRAGE ********************************************************
0014
0015 % taille fenêtre caméra
0016 width=640  ; height=240;
0017
0018 % resizing pour alléger traitement
0019 resizeP = .5;
0020
0021 %paramètres
0022 tol=.1;       % taille de classe
0023 nBinsDem=80   % nombre de bins
0024
0025  % couleur affichage
0026 rectColor = [0 255 0];  % cadre de localisation
0027 classeBidon = [0 1 0 ]; % classe d'exclusion
0028 swColor = [0 0 255 ];   % fenêtre candidate
0029
0030 % taille de la bordure de la fenêtre candidate (avec ses bords)
0031 xBord=8
0032 yBord=8
0033
0034 % facteur d'échelle  (initial = 1)
0035 r=1
0036
0037 % pas d'optimisation du alpha (pour analyse famille d'indicateur)
0038 dAlpha = .01
0039
0040 % Définit intervalle autour de l'axe de symétrie vertical
0041 % dans lequel le robot peut continer à avancer en ligne droite
0042 tolXPos = 40
```

```
0043
0044 % valeur a partir de laquelle on est assez proche
0045 rMax = 1.3
0046
0047 % Combien de capture a prendre entre deux commandes ?
0048 loopAdjust = 2
0049
0050 %% CAMERA ************************************************************
0051 % Constante pour sélectionner l'opération à effectuer
0052 ARG_P_POS = 0;
0053 ARG_P_FRAMERATE = 1;
0054 ARG_P_HARDWAREGAIN_ALL = 2;
0055 ARG_P_CONTRAST = 3;
0056 ARG_P_BRIGHTNESS = 4;
0057 ARG_P_GAMMA = 5;
0058 ARG_P_SIZE = 6;
0059 ARG_P_EXPOSURETIME =7;
0060 ARG_P_MODE=8;
0061 SYNC_FC = 10;
0062 GET_FC = 11;
0063 ARG_P_PCLK  = 12;
0064 MODE_RAW8=0;
0065 MODE_RGB24=1;
0066 IGNORE_PARAM=25; % permet d'ignorer un paramètre (cfr SDK)
0067
0068 %% initialisation caméra, récupération handler
0069 hCam=uEyeMat();
0070
0071 % taille d'image et fenêtrage matériel
0072 uEyeMat(hCam, [ARG_P_SIZE  width height 0 0 ]);
0073 uEyeMat(hCam, [ARG_P_POS 0 120 0 0 ]);
0074
0075 % configuration des gains
0076 uEyeMat(hCam, [ARG_P_HARDWAREGAIN_ALL 15 IGNORE_PARAM IGNORE_PARAM  0 ]);
0077 uEyeMat(hCam, [ARG_P_HARDWAREGAIN_ALL IGNORE_PARAM 12 IGNORE_PARAM  0 ]);
0078 uEyeMat(hCam, [ARG_P_HARDWAREGAIN_ALL IGNORE_PARAM IGNORE_PARAM 37  0 ]);
0079
0080 % correction gamma
0081 %uEyeMat(hCam, [ARG_P_GAMMA 100 0 0 0 ]);
0082
0083 % pixel clock et temps d'exposition
0084 uEyeMat(hCam, [ARG_P_PCLK 5  0 0 0 ]);
0085 uEyeMat(hCam, [ARG_P_EXPOSURETIME 60 0 0 0 ]);
0086
0087 %% INTERFACE ************************************************************
0088
0089
0090 % acceleration de l'affichage
0091 hFig=figure('name','TEST dynamique : sigma(i,alpha_i)');
0092 set(hFig,'Renderer','OpenGL');
0093
0094
0095
0096 %% ACQUISITION DE LA CIBLE **************************************************
0097
0098 disp('Cadrage de la référence')
0099 tic
0100 for i = 1 : 75
0101     z=imresize(uEyeMat(hCam),resizeP);
```

```
0102     subplot(1,1,1),    imshow(z);
0103     title('Caméra : sélection référence');
0104     drawnow;
0105 end
0106 toc
0107 ref=imcrop(z);
0108
0109 % taille de la référence
0110 [mR,nR,bidonR] = size(ref)
0111 disp('Taille référence')
0112 mR*nR
0113
0114
0115 %% INITIALISATION ALGO/INTERFACE ****************************************
0116
0117 % creation de la map sur base de la reference
0118 disp('Creation de la map sur la référence ...');
0119 tic ; [refIndex,map] = rgb2ind(ref,nBinsDem,'nodither'); toc;
0120
0121 % indexation de la reference et de la scene
0122 % en utilisant une classe d'exclusion
0123 disp('Indexation de la reference ...');
0124 tic ; refIndex = img2ind64(ref,map,tol); toc;
0125
0126
0127 % ajout de la classe d'exclusion à la map
0128 map(end+1,:)=classeBidon;
0129 [nBins,oBidon]=size(map);
0130 nBins
0131
0132 % histogramme de la référence, besoin dans l'appel au fooprint
0133 hRefIndex = imhist(refIndex,map);
0134
0135
0136 % parametrage des zone de tracé et définition des handlers et sources de données
0137 subplot(2,3,3);
0138 title('Facteur échelle r');
0139 rr(1)=r;
0140 hRR=plot(rr);
0141 set(hRR,'YDataSource','rr');
0142 set(gca,'YLim',[0 3]);
0143 set(gca,'YLimMode','manual');
0144
0145
0146 subplot(2,3,2);
0147 SSS(1)=0;
0148 hSSS =plot(SSS);
0149 set(gca,'XLim',[0 80]);
0150 set(gca,'XLimMode','manual');
0151 set(hSSS,'YDataSource','SSS');
0152 title('Sigma(alpha_k)');
0153
0154
0155 centre = round(resizeP*width / 2);
0156
0157 disp('Interrompre acquisition par CTRL-C')
0158 c=0; % compteur de capture
0159
0160
```

```
0161 %% PROCESSUS D'ACQUISITION / LOCALISATION / COMMANDE ********************
0162 disp('Prêt...') . pause();
0163
0164 while(1)
0165
0166     for b=1 : loopAdjust
0167         c=c+1;
0168
0169         % acquisition de la scène %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
0170         sc=imresize(uEyeMat(hCam),resizeP);
0171
0172         % indexation de l'image de la scène
0173         scIndex = img2ind64(sc);
0174
0175
0176         %% sauvegarde de la capture
0177         scRec(c).img=sc;
0178         %% //sauvegarde de la capture
0179
0180
0181         % calcul de la surface de similarité %%%%%%%%%%%%%%%%%%%%%%%%%%%%
0182         surf = chist14(refIndex,scIndex, r);
0183
0184         %recherche du maximum
0185         [maximum,i] = max(surf);
0186         [maximum,j] = max(maximum);
0187
0188         x=i(j);
0189         % récupération de la SW pour construire l'empreinte
0190         % prendre directement dans l'image indexée !!
0191         % 1 points + dimension définissent le rectangle
0192         swIndex = imcrop(scIndex, [ j-yBord x-xBord round(nR*r)+2*yBord
                                                          round(mR*r)+2*xBord]);
0193         sw      = imcrop(sc     , [ j-yBord x-xBord round(nR*r)+2*yBord
                                                          round(mR*r)+2*xBord]);
0194
0195
0196         % Estimation de la taille %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
0197         % Analyse d'indicateur sigma
0198         recHist = empRec8(swIndex+1,nBins);
0199         [nCadre,bidon]=size(recHist);
0200
0201         % pour recherche du minimum
0202         mmm = inf;
0203
0204         % itération sur le alpha (voir théorie)
0205         % la valeur du minimum est strictement décroissante tant que
0206         % surEstimation > facteur d'échelle réelle
0207         for alpha=1.15 : -dAlpha:  .85
0208
0209             hRefIndexA=round(alpha*r^2*hRefIndex);
0210             mm = inf;
0211
0212             for cadre = floor(.80*nCadre) : nCadre
0213                 ss=sum(abs(hRefIndexA'-recHist(cadre,:)));
0214                 SSS(cadre)=ss;
0215                 if ss < mm
0216                     mm=ss;
0217                 end
```

```
0218            end
0219
0220            if mm<=mmm
0221                mmm=mm;
0222                alphaOpt=alpha;
0223            else
0224                break
0225            end
0226        end
0227
0228        % lissage du facteur d'échelle par moyenne mobile
0229        r= .35*r + .65*    r*(alphaOpt-dAlpha);
0230
0231
0232        % mise à jour interface %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
0233        %  facteur d'échelle
0234        subplot(2,3,3);
0235        title('Facteur échelle r');
0236        rr(c)=r;
0237        xlim([max(c-100,0) c]);
0238        refreshdata(hRR);
0239
0240        %sigma[alpha_k](i)
0241        refreshdata(hSSS);
0242
0243    end
0244
0245    subplot(2,3,4),   subimage(swIndex,map)
0246    title('FET');
0247
0248
0249    % tracé de l'image avec localisation (cadre vert)
0250    for a =  x: x+round(mR*r);
0251        sc(a, j, : ) = rectColor(1,:);
0252        sc(a, j+round(nR*r), : ) = rectColor(1,:);
0253    end;
0254    for a = j:j+nR*r;
0255        sc( x, a, : ) = rectColor(1,:);
0256        sc( x+round(mR*r), a, : ) = rectColor(1,:);
0257    end;
0258
0259
0260    % tracé de l'image avec localisation (bordure bleue)
0261    %         for a =  max(x-xBord,1): x+round(mR*r)+xBord;
0262    %             sc(a, max(j-yBord,1), : ) = swColor(1,:);
0263    %             sc(a, j+round(nR*r)+yBord, : ) = swColor(1,:);
0264    %         end;
0265    %         for a = max(j-yBord,1):j+round(nR*r)+yBord;
0266    %             sc( max(x-xBord,1), a, : ) = swColor(1,:);
0267    %             sc( x+round(mR*r)+xBord, a, : ) = swColor(1,:);
0268    %         end;
0269
0270    subplot(2,3,1),    imshow(sc);
0271    title('Caméra et localisation');
0272
0273
0274    % // COMMANDE ROBOTIQUE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
0275    if r>rMax
0276
```

```
0277         % Suffisament proche, mettre moteur en float
0278         rcx2b([0 0]);
0279
0280     else
0281         xPos = round(j+(nR*r/2))
0282
0283         fRot = r / rMax;
0284
0285         if xPos<centre-tolXPos
0286             % Barre à tribord !
0287             rcx2b([0,round(fRot*35)]);
0288
0289         elseif xPos>centre+tolXPos
0290             % Barre à babord !
0291             rcx2b([round(fRot*35),0]);
0292
0293         else
0294             % En avant toute !
0295             rcx2b([50,50]);
0296         end
0297
0298     end
0299
0300     % attendre car RCX légo ne suit pas !
0301     pause(.05)
0302
0303     % display refresh
0304     drawnow;
0305
0306 end
```

Annexe B

Cercles de confusions

Format du capteur	Diamètre du cercle de confusion (c, en mm)
Photos 24x36	0,033
Vidéo 1"	0,030
Vidéo 2/3 "	0,021
Vidéo 1/2 "	0,016
Vidéo 1/3 "	0,011

Annexe C

Comparaison des performances

Nous allons essayer de voir l'impact réel des améliorations différentielles et SSE2 sur le temps nécessaire au calcul du CHS. Si la complexité des deux algorithmes est d'ordre polynomial, le degré de celle de l'algorithme optimisé est cependant plus faible. La différence n'est pas très marquée en termes de complexité, mais il en résulte cependant que l'algorithme optimisé est applicable au temps réel tandis que l'autre, non. Afin de les comparer, nous allons donc tenter d'estimer le nombre d'opérations simples nécessaires à chacun de ces algorithmes.

Supposons que la fenêtre de recherche soit de taille $m * n$, que la scène acquise soit de taille $O * P$ et que la palette de couleurs utilisée compte k couleurs.

Dans un premier temps nous allons détailler le nombre d'opérations nécessaires aux différentes phases de la version intuitive du calcul du CHS.

- Le nombre d'opérations nécessaires au calcul de chacun des histogrammes est égal au nombre de pixels dans la fenêtre de recherche :

$$m * n$$

- Le nombre d'opérations requises pour une intersection d'histogrammes est donné par le nombre de bins à intersecter multiplié par le nombre d'additions correspondant au temps d'exécution d'un minimum, augmenté du nombre de bins, car il faut additionner les valeurs des k intersections :

$$k * Tmin + k$$

  (Notre estimation du temps nécessaire pour effectuer un minimum, Tmin, correspond à celui de trois additions.)

- Le nombre de positions pour la fenêtre de recherche est, quant à lui, donné par :

$$(O - m + 1) * (P - n + 1).$$

Pour l'algorithme intuitif, nous avons donc $(m * n) * (O - m + 1) * (P - n + 1)$ opérations pour le calcul des histogrammes et $(O - m + 1) * (P - n + 1) * (k * Tmin + k)$ opérations pour le calcul des intersections d'histogrammes.

Nous allons maintenant estimer le nombre d'opérations nécessaires au calcul du CHS pour l'algorithme optimisé qui utilise les instructions SSE2.

- Le nombre d'opérations simples pour le calcul du premier histogramme est donné par :

$$m * (n + 7) * plusSSE2,$$

où $plusSSE2$ correspond au nombre d'opérations simples équivalentes en temps d'exécution à une addition SSE2, soit 2 selon nos approximations. Nous avons ici $n + 7$ au lieu de n, car la version optimisée calcule 8 histogrammes simultanément et utilise, par conséquent, une fenêtre de recherche à laquelle 8 lignes de pixels ont été rajoutées.

- Le nombre de décalages verticaux de la fenêtre de recherche est donné par $\frac{P-n+1}{8}$, car un décalage vertical correspond à un saut de 8 lignes.
- Le nombre d'opérations pour le passage d'une ligne à la suivante est donné par le nombre de pixels à retirer et à rajouter sur l'histogramme, soit m pour la première ligne de la fenêtre de recherche et m pour la dernière. Puisqu'un saut de lignes compte 8 lignes, celui-ci nécessite $16 * m * plusSSE2$ opérations. La formule suivante détermine le nombre total d'opérations indispensables à l'ensemble des sauts de lignes :

$$(P - n + 1) * m * 2 * plusSSE2$$

- Le nombre d'opérations nécessaires pour un décalage de colonne de la fenêtre de recherche est donné par $2*(n+7)*plusSSE2$. Le nombre total d'opérations pour l'ensemble des décalages de colonne est donc donné par :

$$(O - m + 1) * plusSSE2 * \frac{P - n + 1}{8} * (n + 7) * 2$$

- Le nombre d'opérations nécessaires pour une intersection d'histogrammes est obtenu de la même manière que celle utilisée pour l'algorithme intuitif :

$$k * TminSSE2 + k$$

- Le nombre de positions de la fenêtre de recherche SSE2 pour lesquelles il faudra calculer l'histogramme explicitement sera 8 fois moins élevé, puisque les intructions SSE2 nous permettent de calculer 8 positions simultanément. Ainsi, nous aurons $(O - m + 1) * \frac{P-n+1}{8}$ positions à calculer.

Le nombre total d'opérations nécessaires au calcul des différents histogrammes est donné par :

$$\begin{gathered} m * plusSSE2 * ((n + 7) + (P - n + 1) * 2) \\ + \\ (O - m + 1) * \frac{P-n+1}{8} * (plusSSE2 * (n + 7) * 2) \end{gathered}$$

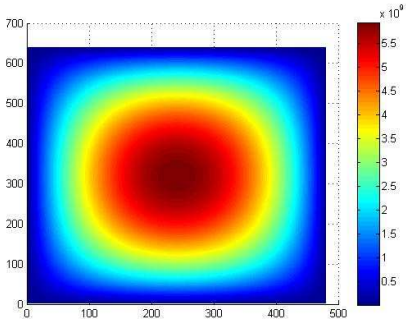
Le nombre d'opérations pour les intersections d'histogrammes est, lui, donné par :

$$(O - m + 1) * \frac{P - n + 1}{8} * (k * TminSSE2 + k).$$

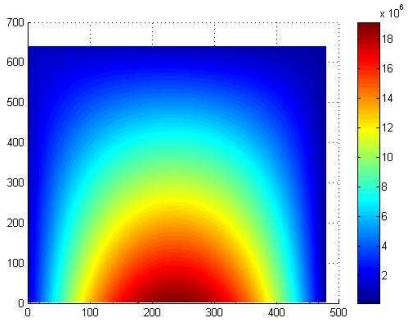
En prenant l'exemple d'une scène de 640 pixels par 480, le nombre d'opérations simples nécessaires est de l'ordre de 10^9 pour l'algorithme non optimisé, et de l'ordre de 10^7 pour la version optimisée. Les optimisations apportées tant au calcul des histogrammes qu'à celui de leurs intersections sont illustrées sur la figure [C.1]. Dans sa version non optimisée, l'algorithme n'est que peu influencé par le nombre de bins de l'histogramme. Cependant, celui-ci devient plus critique dans la version optimisée. En effet, dans la partie concernant l'intersection d'histogrammes, seule l'utilisation des opérations SSE2 intervient. Avec un nombre élevé de classes, le nombre d'opérations nécessaires aux intersections d'histogrammes peut devenir proche du nombre total d'opérations à effectuer dans la méthode optimisée et, de cette façon, constituer un facteur critique. Ce phénomène est visible sur la figure [C.1]. Nous constatons que le calcul des histogrammes totalise au maximum $2 * 10^6$ opérations tandis que l'intersection des histogrammes peut prendre jusqu'à $7 * 10^7$ opérations pour les fenêtres de recherche de petite taille. La différence d'allure entre les surfaces de la méthode optimisée pour 6 bins (Figure[C.2(e)]) et pour 256 bins (Figure[C.2(f)]) illustre elle aussi ce phénomène.

En ayant pris comme hypothèse que nous travaillons sur des scènes en 640 * 480, nous avons calculé le rapport entre le nombre d'opérations nécessaire pour les 2 algorithmes avec respectivement 6 et 256 classes de couleurs. Les résultats obtenus sont illustrés dans la figure[C.2]. Les valeurs en abscisse et en ordonnée de ces graphiques représentent respectivement la hauteur et la largeur de la fenêtre de recherche utilisée, et la couleur du point correspondant nous donne une estimation du nombre d'opérations simples à effectuer.

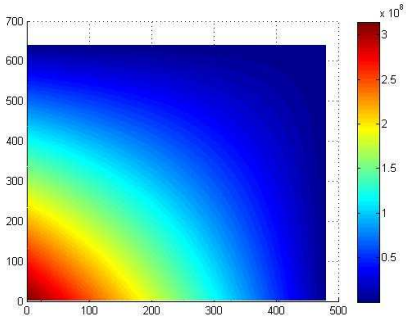
Il est important de noter que seuls les bins de couleurs présents dans la référence seront pris en compte ; les autres, afin de limiter les calculs, sont ignorés par l'algorithme optimisé. Si 6 bins sont utilisés dans la référence, nous constatons sur la figure[C.2] que pour plus de 90 % des tailles de fenêtre de recherche possibles, il faut 100 fois moins d'opérations. Avec 256 bins utilisés dans la référence, les améliorations sont moindres. Nous constatons que pour 90 % des tailles de fenêtre de recherche possibles, l'algorithme nécessite 10 fois moins d'opérations et il en nécessite cent fois moins pour seulement 60 %. Cependant, les tailles de fenêtre de recherche où le gain est moindre correspondent à celles pour lesquelles le nombre de calculs dans la version non optimisée est moindre.



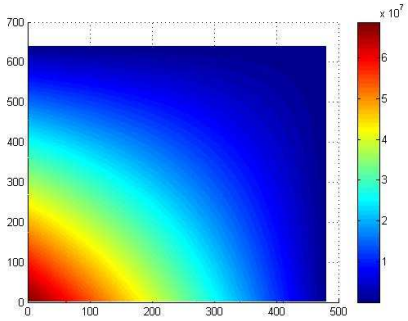
(a) Calcul des histogrammes version non optimisée



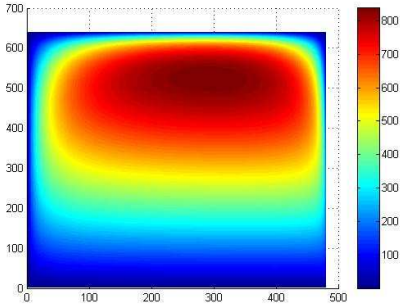
(b) Calcul des histogrammes version optimisée



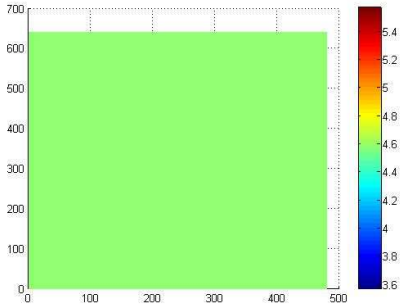
(c) [Intersection des histogrammes non optimisée (256



(d) Intersection des histogrammes optimisée (256 bins)

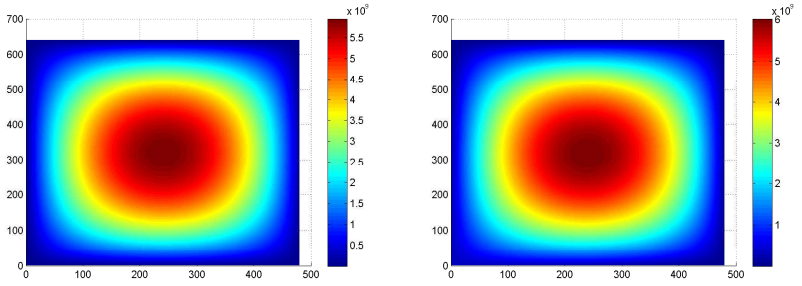


(e) Comparaison pour le calcul des histogrammes

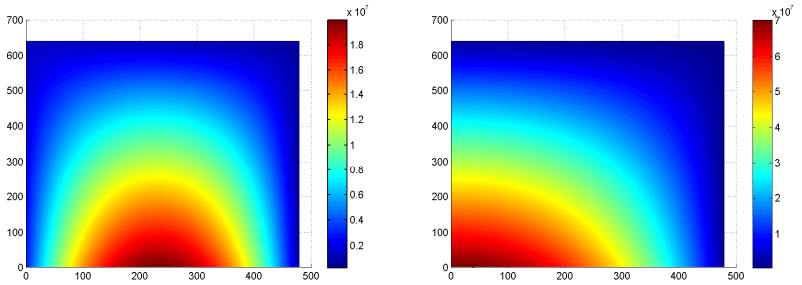


(f) Comparaison pour l'intersection des histogrammes

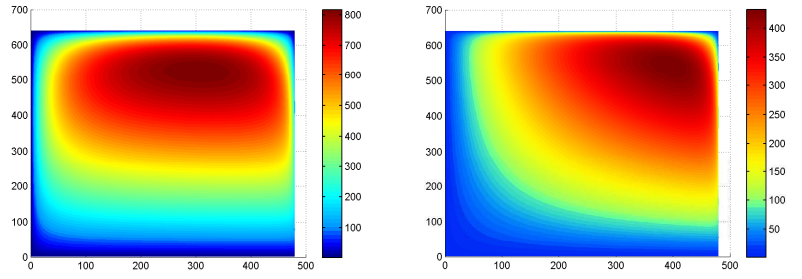
FIG. C.1 – Comparaison du nombre d'opérations simples entre les algorithmes optimisés et non optimisés pour une image de 640 pixels par 480 pour le calcul et l'intersection des histogrammes, avec une palette de 256 bins.



(a) Algorithme simple avec 6 bins (b) Algorithme simple avec 256 bins



(c) Algorithme optimisé avec 6 bins (d) Algorithme optimisé avec 256 bins



(e) Comparaison pour 6 bins (f) Comparaison pour 256 bins

FIG. C.2 – Comparaison du nombre d'opérations simples entre les algorithmes optimisés et non optimisés pour une image de 640 pixels par 480, avec une palette de 6 bins et une palette de 256 bins.

Annexe D

LST

Dans cette annexe, nous allons dans un premier temps montrer pourquoi la définition de la saturation n'est pas ambiguë. Ensuite, nous parlerons des alignements de points et illustrerons ces phénomènes. Nous mettrons aussi en évidence les points qui nous semblent suspicieux.

La définition de la saturation de LST n'est pas ambiguë Contrairement à ce que pourrait laisser penser la définition donnée dans la section 3.5.1, il n'y a pas d'ambiguïté pour la définition de la saturation lorsque $m = med(r, g, b)$. En effet,

$$m = \frac{r + g + b}{3}$$

Or, la médiane de trois éléments est la valeur intermédiare de ces trois éléments. Cette équation est donc strictement équivalente à :

$$m = \frac{max(r, g, b) + med(r, g, b) + min(r, g, b)}{3}$$

Par conséquent :

$$med = \frac{max(r, g, b) + med(r, g, b) + min(r, g, b)}{3}$$

Ce qui se réécrit :

$$med(r, g, b) = \frac{max(r, g, b) + min(r, g, b)}{2}$$

$$\frac{3}{2} * \left(max(r, g, b) - \frac{max(r, g, b) + min(r, g, b)}{2} \right) = \frac{max(r, g, b) - min(r, g, b)}{2}$$

$$\frac{3}{2} * \left(\frac{max(r, g, b) + min(r, g, b)}{2} - min(r, g, b) \right) = \frac{max(r, g, b) - min(r, g, b)}{2}$$

Lorsque $m = med$, nous avons donc égalité entre les deux définitions de la saturation.

Découverte de l'espace LST et de ses propriétés. Afin de disposer d'une représentation polaire de la couleur conduisant à des mesures de brillances et de saturations aux propriétés cohérentes, Serra à introduit la représentation LST dans laquelle la norme L_1 est utilisée pour définir la saturation et la luminosité. La définition complète de cet représantion a été donnée dans la section 3.5.1.

Nous nous sommes intéressés à cette représentation car elle permet de détecter des zones d'ombres et de reflets dans un image, qui sont des zones pour lesquelles les couleurs perçues par la caméra sont fortements dénaturées. Il semble donc intéresant de pouvoir détecter ces zones, afin de ne pas en tenir compte ou d'envisager un traitement spécifique à celles-ci pour retrouver leurs couleurs originelles.

Afin de visualiser les propriétés de cet espace couleur, nous allons utiliser deux images, la première représentant une table de guitare et la seconde une bouée(Figure D.1).

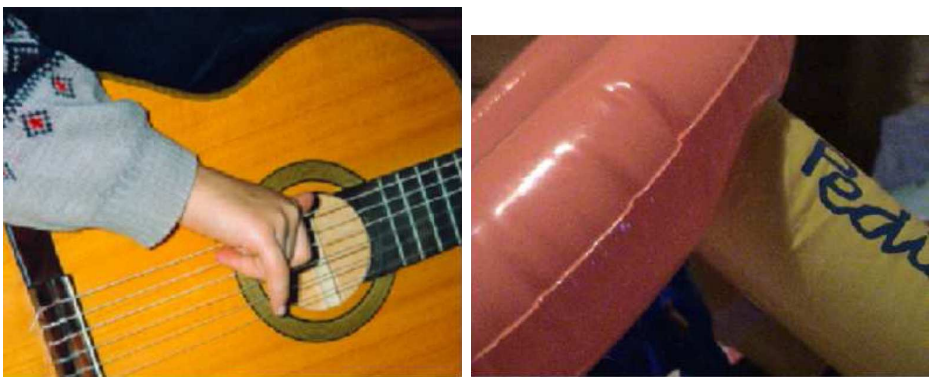
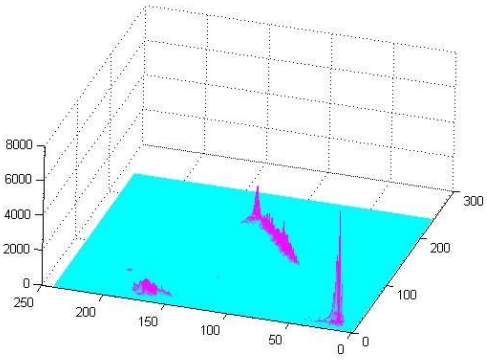


FIG. D.1 – Image de guitare et de bouée illustrant les phénomènes d'ombres et de reflets (Soure [26]).

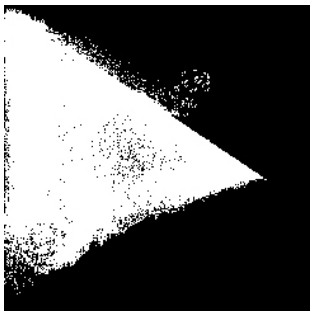
Afin de détecter ces zones où la lumière à des propriétés physiques particulières, il faut élaborer l'histogramme bidimensionnel L/S de l'image (ici la guitare). Sur la figure D.2, plusieurs représentations de cet histogramme sont représentées. La première représente l'histogramme sous forme de surface, la seconde sous forme d'image binaire, indiquant simplement si oui ou non il y a des pixels dans ce bin de l'histogramme et, enfin, la dernière, en niveau de gris, représentant les proportions de pixels dans chacun des bins de cet histogramme L/S. Dans l'histogramme bidimensionnel L/S, sous forme d'image en niveaux de gris, des alignements de points apparaissent (ces alignements sont aussi visibles sur la surface représentant l'histogramme). Serra a montré dans [26] que ces alignements peuvent correspondre à trois types de régions particulières, à savoir :

- des ombres de teinte fixe (ombre portée de la bouée) ;
- des dégradés de lumière sur un plan (table de la guitare) ;
- des reflets non complètement saturés (doigt de la main sur la guitare) ;

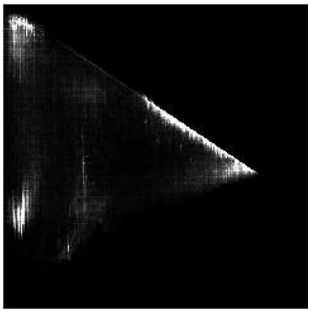
Les figures D.3(a) et D.3(b) montrent les alignements présents pour, respectivement, l'image guitare et l'image bouée ainsi que les régions auquelles correspondent ces alignements. La plupart des alignements passent soit par le point (0,0) du plan



(a) Sous forme de surface



(b) Sous forme d'image binaire



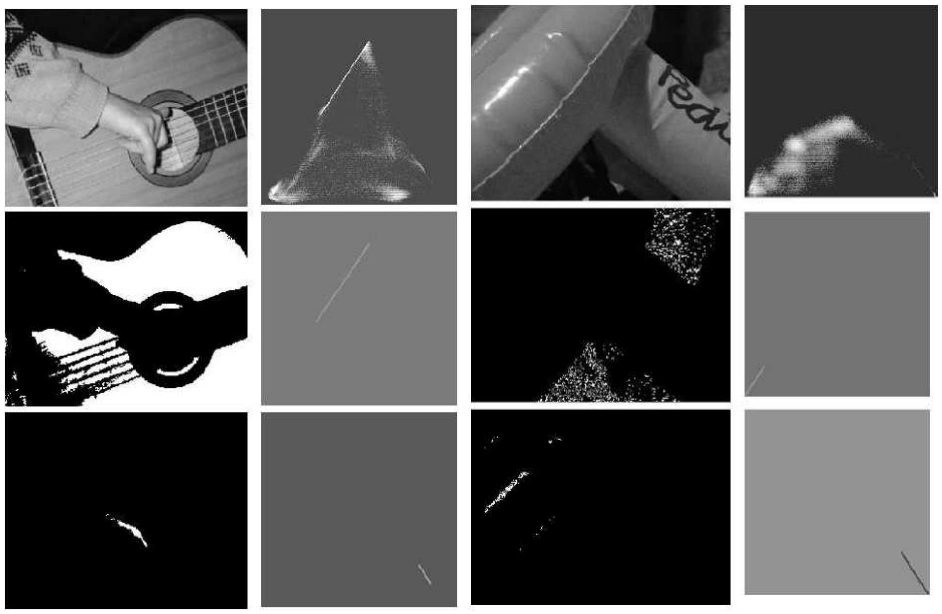
(c) Sous forme d'image en niveau de gris

FIG. D.2 – Différentes représentation de l'histogramme bidimentionnel

luminance/saturation (la table de guitare et l'ombre portée de la bouée), soit par le point (1,0) de ce plan (les alignements correspondant aux reflets). Dans [26], Serra et Angulo ont établi un modèle théorique pour expliquer ces phénomènes d'alignements de points.

Afin de voir si cet espace peut nous apporter des résultats, nous avons implémenté un librairie en C utilisable depuis Matlab afin de convertir les images RGB dans cet espace couleur. Nous avons également créé des tests pour vérifier ces propriétés. Les alignements visibles dépendent de la manière dont nous créons l'image en niveau de gris. Si l'intensité des pixels dépend de la proportion de cette classe de pixels, les petites zones présentant des ombres ou des reflets n'apparaîtrons pas comme étant des alignements de points. Comment mettre en évidence tous les alignements de points ?

D'autre part, Angulo rajoute que pour chaque alignement de pixels, la teinte garde une valeur constante. Pourtant, selon la définition de la teinte et de la saturation, un pixel de couleur rouge pure appartiendra au même alignement qu'un pixel de couleur bleue pure. En effet, les définitions de ces grandeurs utilisent la norme L_1 et sont donc symétriques par Nous nous attendions donc a trouver des images dans lesquelles des allignements ne correspondent pas à des zones de teinte identique. Si ce phénomène a facilement été mis en évidence sur des images construites artificiellement,



(a) Image de guitare et ses alignements (b) Image de bouée et ses alignements

FIG. D.3 – Alignements de points dans les images guitares et bouées (Soure [26])

nous n'avons pu l'établir sur des images réelles, et ce malgré de nombreux tests notament sur des images de la BSDB (Berkeley Segmentation Dataset and Benchmark).

Cependant, nous avons aussi remarqué que le plupart des alignements mis en évidences par ces histogrammes correspondent en fait aux frontières de l'histogramme bidimensionnel, fronctières qui sont formées par des pixels dont les valeurs RGB ont une/des composante(s) saturée(s) ou nulle(s) (Figure D.4).

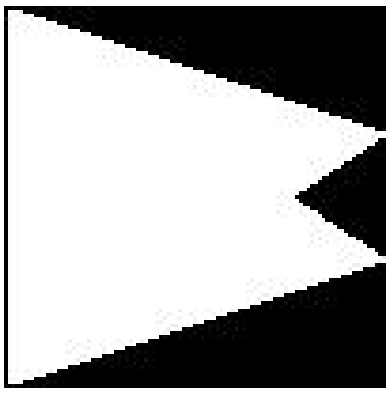


FIG. D.4 – Représentation du cube RGB dans l'espace LS (les pixels blanc représent la partie du plan LS dans lequel se trouve l'ensemble des valeurs RGB).

Nous pouvons, pour ces alignements, obtenir les mêmes zones de l'image en n'utilisant que les composantes RGB, sans avoir à effectuer tout ce supplément de calcul.

Notre analyse de cet espace nous a confronté à des résultats pour le moins mitigés,

et nous n'avons pas poussé plus en avant nos investigations.

Annexe E

Résultats pour la recherche de motifs

La première série d'images (Figure E.1) est constituée d'images synthétiques simples permettant de valider la méthode utilisée pour estimer la taille et l'orientation de la cible. Chaque image présente ci-dessous est constituée de deux figures contenant la même imagette. Le résultat du suivi des motifs est présenté dans la figure de droite tandis que sur la figure de gauche l'on a reporté la position, la taille et l'orientation estimée de la cible par le biais d'un rectangle vert. Les motifs ont étés sélectionnés manuellement sur la première image de la séquence. Un des motifs est suivi au moyen d'un rectangle cyan, l'autre au moyen d'un rectangle magenta.

Dans la seconde (Figure E.2), il s'agit d'images réelles, et plus particulièrement d'images extraites d'une séquence vidéo réalisée par Urban. Nous avons appliqué l'algorithme de suivi de motifs sur cette séquence. Nous avons utilisé trois motifs, sélectionnés manuellement, pour suivre le bâton de colle. Dans cette série aussi, l'image de référence et ces motifs ont étés sélectionnés à partir de la première image de la séquence. Le rectangle vert de la figure de gauche identifie l'objet dans la scène au moyen des estimations de taille et d'orientations relevés à l'itération précédente. La figure de droite est à présent restreinte au voisinage de la cible identifiée, et les motifs y sont recherchés afin d'ajuster l'estimation de taille et d'orientation. Le résultat de ces estimations est reporté sur la figure de gauche par le biais du rectangle jaune.

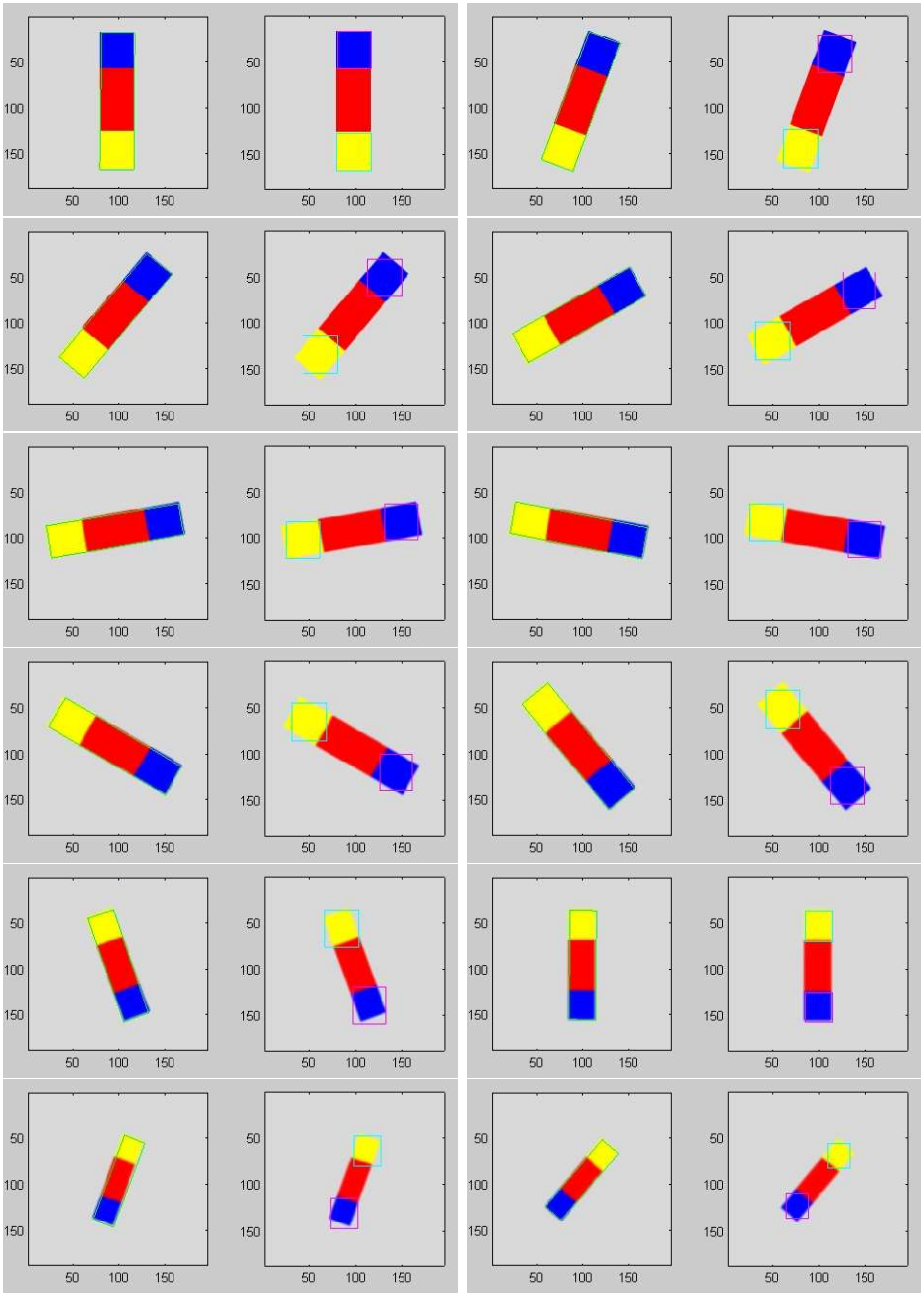


FIG. E.1 – Application du suivi de motifs sur une première série d'images.

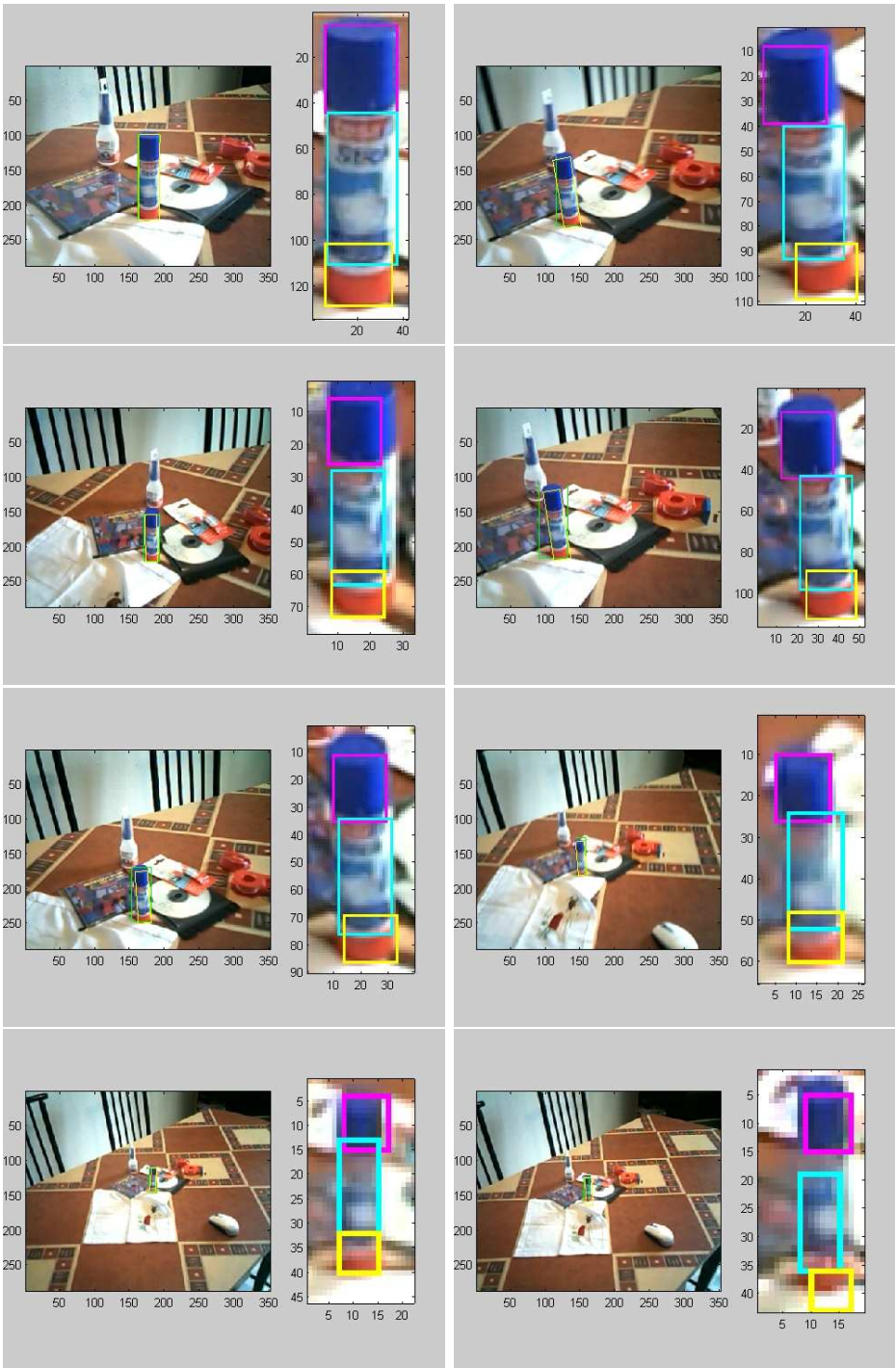


FIG. E.2 – Application du suivi de motifs à une séquence d'images réelles.